NASA CONTRACTOR
REPORT

NASA CR-150901

PLACEMENT TECHNIQUES FOR THE STANDARD TRANSISTOR ARRAY (STAR)

By G. W. Cox and B. D. Carroll
Electrical Engineering Department
Auburn University
Auburn, Alabama 36830

September 11, 1978

FOR EARLY DOMESTIC DISSEMINATION

Because of its significant early commercial potential, this information, which has been developed under a U. S. Goverment Program, is being disseminated with the United States in advance of general publication. This information may be duplicated and used by the recipient with the express limitation that it not be published. Release of this information to other domestic parties by the recipient shall be made subject to these limitations. Foreign release may be made only with prior NASA approval and appropriate export licenses. This legend shall be marked on any reproduction of this information in whole or in part.

Date for general release: September 1980

Prepared for

NASA- GEORGE C. MARSHALL SPACE FLIGHT CENTER
Marshall Space Flight Center, Alabama 35812

TECHNICAL REPORT STANDARD TITLE PAGE

1. REPORT NO.	2. GOVERNMENT ACCESSION NO.	3. RECIPIENT'S CATALOG NO.
NASA CR-150901		

4. TITLE AND SUBTITLE	5. REPORT DATE
Placement Techniques for the Standard Transistor Array (STAR)	September 11, 1978
	6. PERFORMING ORGANIZATION CODE

7. AUTHOR(S)	8. PERFORMING ORGANIZATION REPORT #
G. W. Cox and B. D. Carroll	

9. PERFORMING ORGANIZATION NAME AND ADDRESS	10. WORK UNIT NO.
Electrical Engineering Department Auburn University Auburn, Alabama 36830	11. CONTRACT OR GRANT NO. NAS8-31572
	13. TYPE OF REPORT & PERIOD COVERED Contractor Report
12. SPONSORING AGENCY NAME AND ADDRESS National Aeronautics and Space Administration Washington, D. C. 20546	14. SPONSORING AGENCY CODE

15. SUPPLEMENTARY NOTES

This work was done under the technical supervision of Mr. Jack Matheney, George C. Marshall Space Flight Center, Alabama.

16. ABSTRACT

Techniques for the placement of digital logic cells on STAR's (Standard Transistor Arrays) have been developed. These techniques are designed to provide placement automation capabilities to low-volume users with limited available processing power.

A series of programs implementing the placement techniques has been written and a number of test cases executed. Results of these test cases indicate that the methods achieve near-optimum placements with respect to network routability.

17. KEY WORDS	18. DISTRIBUTION STATEMENT
	For early domestic dissemination F. Brooks Moore F. BROOKS MOORE, Director Electronics and Control Laboratory, MSFC

19. SECURITY CLASSIF. (of this report)	20. SECURITY CLASSIF. (of this page)	21. NO. OF PAGES	22. PRICE
Unclassified	Unclassified	103	NTIS

ACKNOWLEDGMENT

The authors thank Mr. John Gould and Mr. Bob Jones of Marshall Space Flight Center who developed the clustering and linear placement procedures and who provided constructive suggestions for the STAR placement techniques described herein. This work was supported by the National Aeronautics and Space Administration - Marshall Space Flight Center under contract NAS8-31572.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

I. INTRODUCTION

The advent of semicustom digital integrated circuits is indicated by the increasing number of manufacturers supplying these devices [1]. In general, the semicustom digital IC approach uses a standard array of electronic components which require only metal interconnection for customization. Among the advantages of this approach are:

- Design secrecy,
- Potential reliability gain due to reduced chip and interconnection counts,
- Power reduction, and
- Size reduction.

While the true custom integrated circuit approach (requiring customization of diffusions as well as interconnections) shares these advantages, the typical cost and quantity requirements may be prohibitive to a low-volume user.

The Standard Transistor ARray (STAR) developed at Marshall Space Flight Center [2] is an approach to low-cost semicustom IC development. This system uses a predefined array of transistors with a comprehensive library of standard digital logic elements (cells) to achieve ease of circuit implementation. Internal cell connections are

standardized so that the user need not be concerned with the actual devices, but only with cell interconnections.

While manual layout and routing of cells on a STAR is possible, handling of large circuits can become complex. Thus, the design of an automated placement and routing system has been undertaken. The organization of the layout portion of this system is described in this report. The remaining portions of this section will describe the system objectives and STAR modelling. Later sections will outline the placement procedures, their implementation, and experimental results.

PLACEMENT OBJECTIVES

The following considerations dictate the objectives of the placement procedure:

1. Since large computing systems may not be available to many potential system users, the automated procedures should be adaptable to relatively inexpensive computers with limited processing power and graphics capability. In addition, the procedures should execute and provide results as quickly as possible.

2. The STAR consists of a row-column organized array of complementary MOS transistors with a cell residing entirely on one double transistor row. A typical STAR cell is shown in Figure 1. Two levels of metallization (one for vertical paths and the other for horizontal paths) are used both for internal and external cell connection. Since each direction of routing is, for the

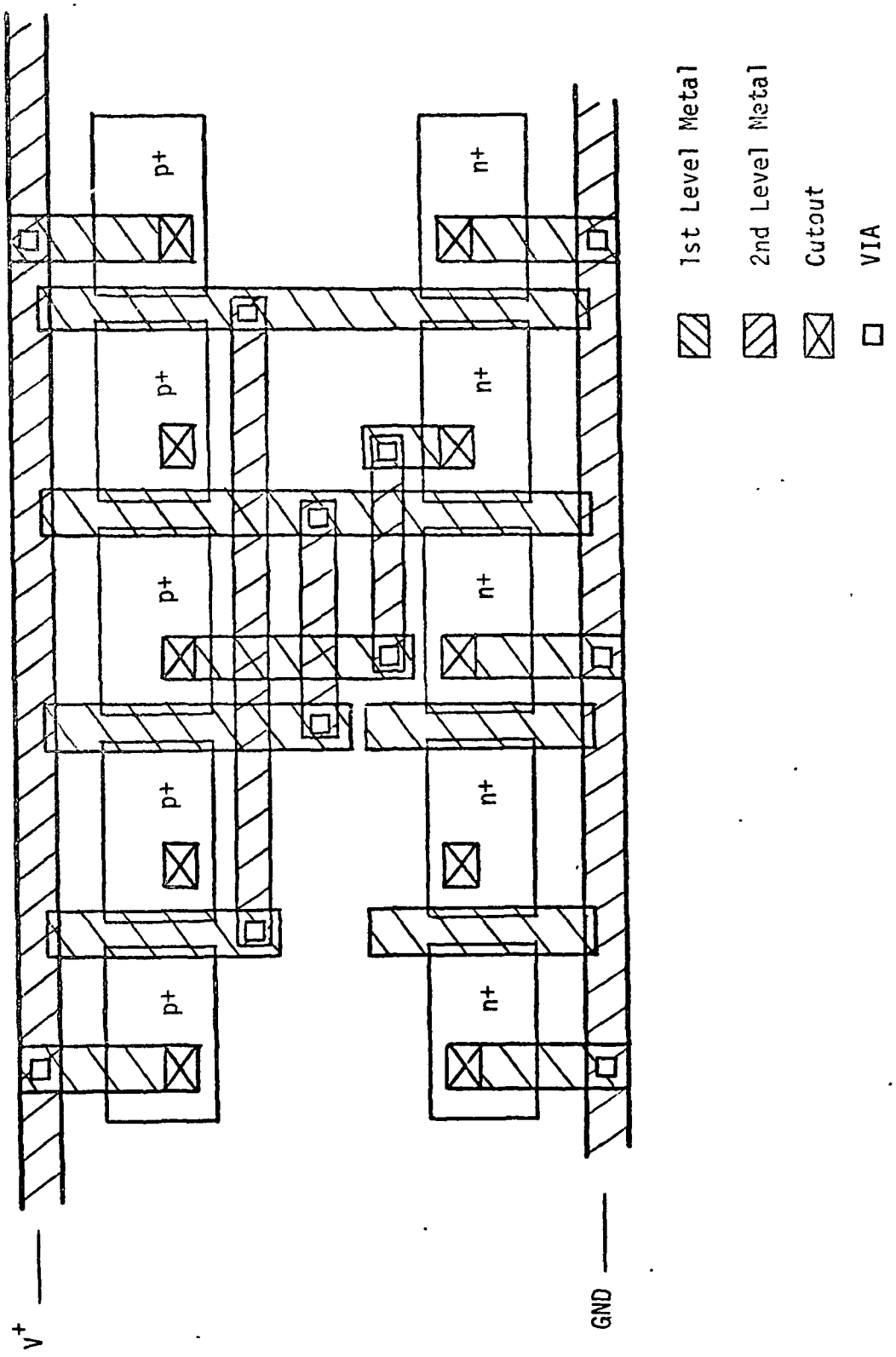


Figure 1. STAR Logic Cell

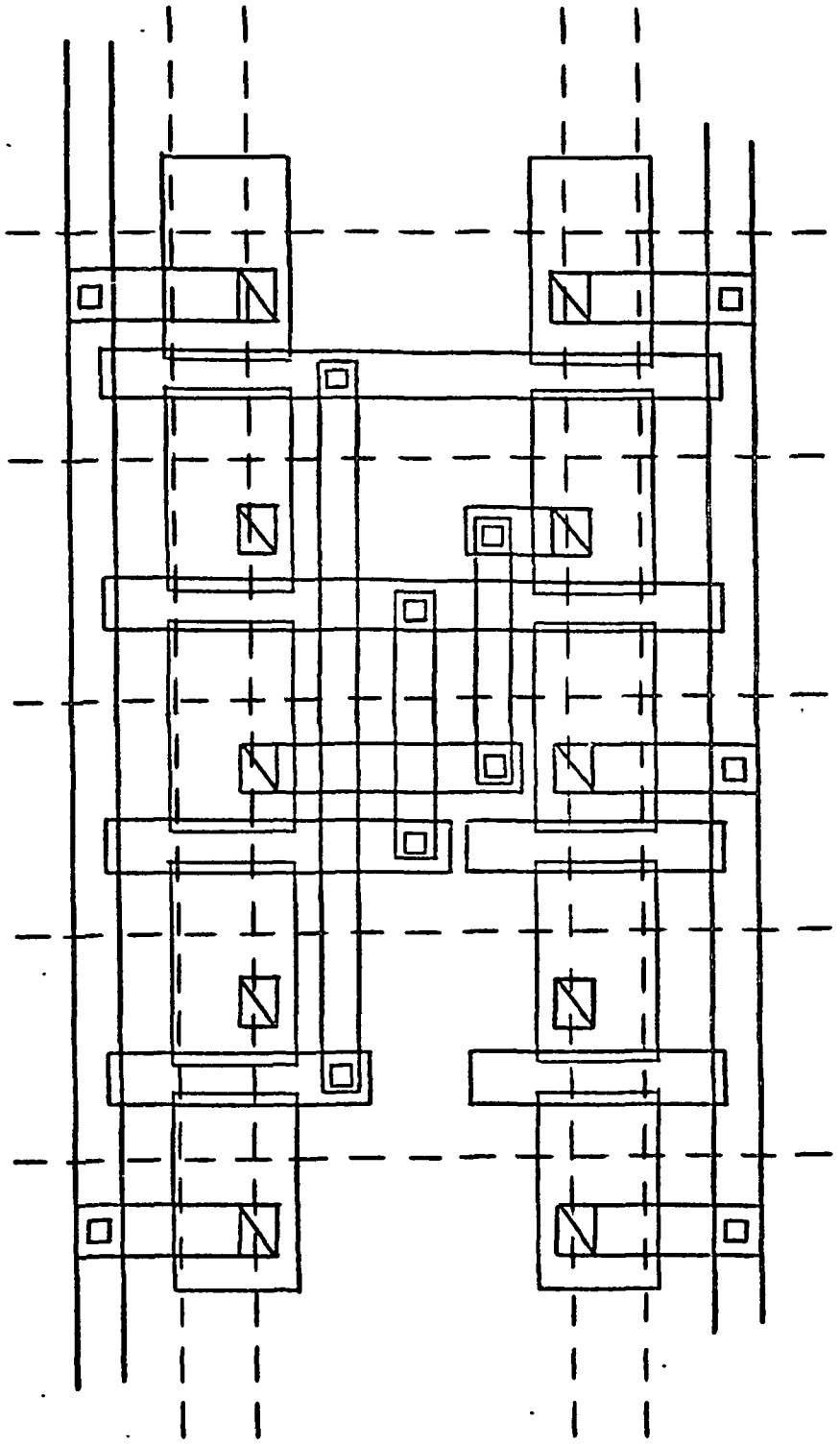
most part, isolated from the other, a number of paths for non-terminating routes exist through a cell. In general, tnere are four of these global paths per double transistor row available in the horizontal direction and one per transistor column in the vertical direction (Figure 2). The placement procedure, then, should produce a cell arrangement which can be routed without violating global path restrictions.

3. Since vertical and horizontal paths are formed on different levels, an interconnection which does not follow a straight line requires a VIA between levels through the insulation oxide. An objective of the placement technique is to hold the number of VIA's required to a minimum by maximizing the number of connections routable as straight lines.

STAR MODELLING

The row-column organization of the transistor array lends itself to a matrix-oriented model of the STAR. For convenience in the placement process, each double row of complementary transistors in the array is called a cell row and is represented as a single row in tne matrix model. An n x m transistor array, then, is modelled with an (n/2) cell row by m transistor column matrix. In this matrix, placed cells occupy a single row and a number of columns across the row. The number of transistor columns occupied by a cell is called the cell's width.

The exact modelling of a cell and its external connections involves the specification of interconnection locations for each cell. The internal structure of a cell, however, is typically bus-oriented and there may be a large



— — — — Global Path Centerline

Figure 2. STAR Logic Cell Global Paths

portion of the cell boundary which can be crossed by a straight connection to any given bus (Figure 3). For placement purposes, then, the entire cell boundary is regarded as the range of all internal connection points and a p-cell net is modelled as straight if:

1. All p cells reside on the same cell row, or

2. There is at least one transistor column passing through all p cells.

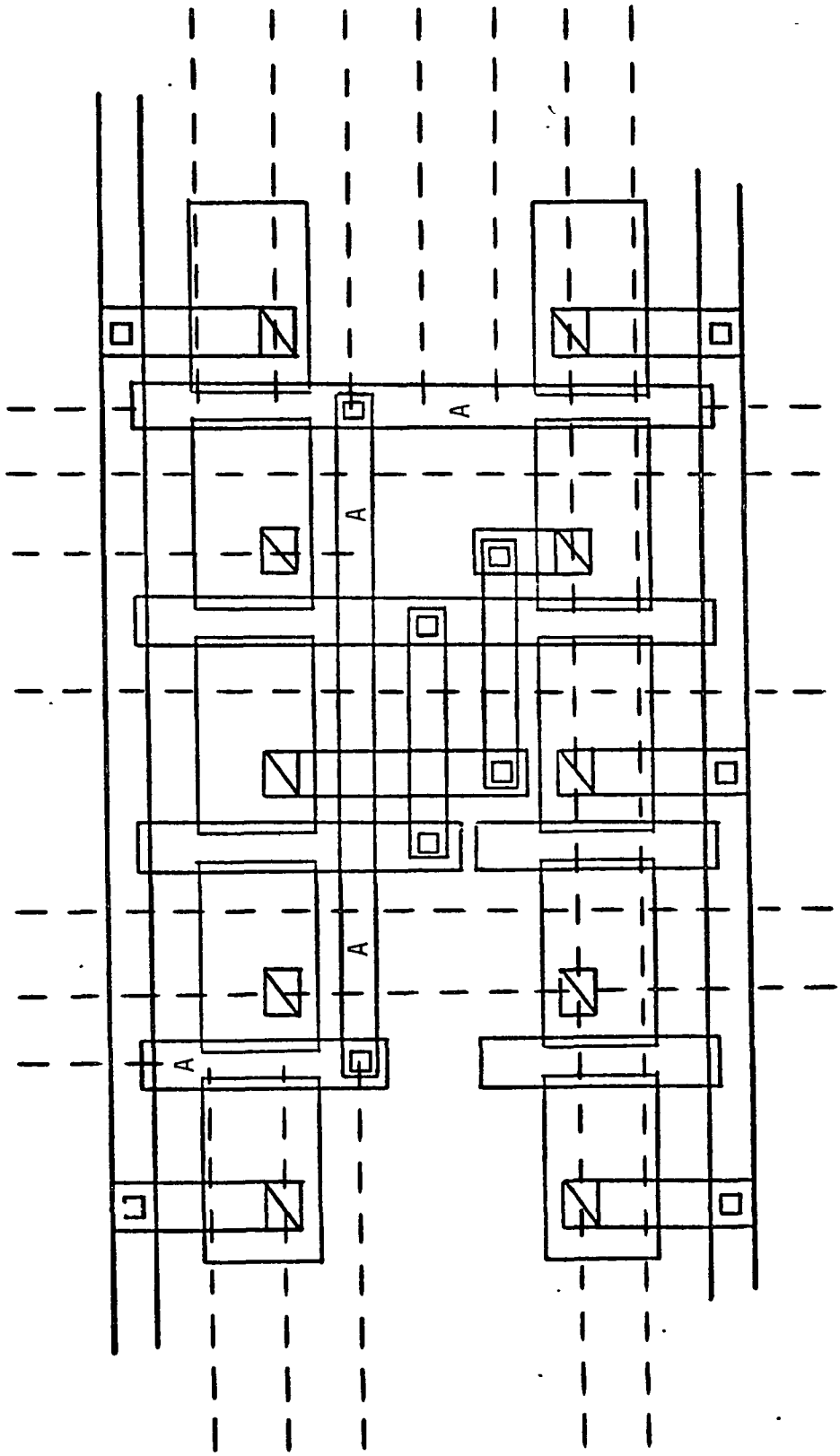


Figure 3. STAR Logic Cell Bus Range

II. PLACEMENT METHODOLOGY

The STAR placement procedure described here may be viewed as being composed of four tasks.

1. The network description task, in which the nets and cells of the network are defined and network definition databases are constructed.
2. The cell clustering task, in which groups of cells are combined based on their interconnection strength.
3. The linear ordering task, in which the formed cell clusters are used to specify a one-dimensional ordering of the cells.
4. The placement task, in which the linear order is mapped onto the 2-dimensional STAR subject to chip size constraints.

A description of each of these tasks will be given in the following sections.

NETWORK DESCRIPTION

For the placement problem, a digital network may be considered as a set of nets interconnecting a set of network gates (cells). The network description task, then, involves specifying, for each net in the system, the cells that the net connects. This information is placed in a list of net and cell numbers called a net list. In addition, it is necessary to include information on network pads and on cell widths in the databases constructed. This information is

used in the succeeding segments of the placement proceedure. An example network is shown in Figure 4, with the pertinent network description shown in Figure 5.

CELL CLUSTERING

The clustering procedure is similar to the method described in [3]. Several considerations are important in the design of this procedure:

1. Due to the structure of the linear ordering segment, the earliest-clustered elements tend to be neighbors in the linear ordering and in the final placement. Experimental results on actual digital networks show that, as a rule, the majority of nets connect only two or three cells. Thus it is desirable to cluster cells in two or three cell nets early in the procedure to provide maximum nearness of these cells in the placement. This can be accomplished by initially considering only cells (or clusters) in two or three cell nets for combination. In the clustering procedure, the variable ALL is used to control clustering by forcing consideration only of nets containing three or fewer cells prior to the formation of ALL clusters.

2. As noted in [3], early formation of large clusters tends to destroy information about smaller clusters since the large clusters typically may be very strongly connected to unclustered cells. A reasonable solution to this problem is the restriction of cluster size to allow a more uniform rate of cluster development. The parameter WMAX is used in the procedure as the cluster size restriction.

3. While computation of a clustering value for all connected pairs of cells (or clusters) at a given stage of cluster development isolates the most profitable pair to cluster in the next stage, it also is very time-consuming for large networks. The approach selected here involves the setting of a clustering

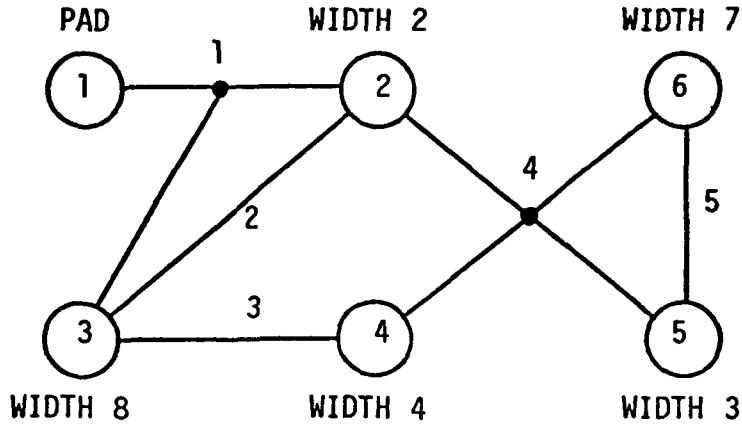


Figure 4. Example Network

NETS:	CELL WIDTHS:	PADS:
1- 1,2,3	1 - 0	1
2- 2,3	2 - 2	
3- 3,4	3 - 8	
4- 2,4,5,6	4 - 4	
5- 5,6	5 - 3	
	6 - 7	

Figure 5. Example Network Description

value threshold, the examination of the clustering value of unclustered connected cells (and clusters), and the combination of those pairs whose value meets the threshold. This method produces results near in quality to the first method, yet requires less processing time. Selection of the clustering threshold, Z7, will be discussed in a later section.

4. The combination of a cell pair depends, for the most part, on the 'connectedness' of the pair. Thus, to achieve 'good' clusterings, a number of decisions regarding interconnection modelling and connectedness computation must be made. Among the factors which must be considered are:

   a) The representation of multi-cell nets, and

   b) The effect of indirect connections between connected cells (cycles).

   The clustering method presented here may be considered as a net absorbing process which requires no special handling of multi-cell nets. The procedure, then, requires no explicit multi-cell net model. Indirect connections are considered on a limited scale by the inclusion of factors regarding their presence in the computation of the clustering value.

5. Clustering size restrictions previously mentioned may produce a final state which is composed of several clusters. Since the linear ordering process (to be described in a later section) requires a single cluster as a starting point, a final clustering step, performed without regard to cluster size, may be necessary.

   Also required by this later step is an ordered record of cluster formation showing the cell or cluster pair used for forming each cluster (the constituents of the cluster). This record will be referred to as the Clustering History Record (CHR). To aid in maintaining order within this list, the first cluster is assigned the number 5Ø1 with succeeding clusters numbered sequentially. The variable C5 is used for cluster counting in the procedure.

CLUSTERING PROCEDURE

The clustering procedure developed will now be described. For convenience, the term cluster will be used to describe both individual cells and cell clusters.

1. Set ALL (number of clusters to be formed using two or three cluster nets), WMAX (maximum cluster size), and Z7 (clustering value threshold). C5=0.

2. Select the next net in the net list which is not completely contained in a cluster. If the list is exhausted, go to step 8.

3. If C5 < ALL and the net contains more than three clusters, repeat step 2.

4. Pick 2 clusters, C1 and C2, in the selected net for trial combination such that the size of the resulting cluster would be less than or equal to WMAX. If none exist, go to 2.

5. Compute the clustering value of C1 and C2.

6. If the computed clustering value is less than Z7, repeat step 4.

7. C5=C5 + 1. Combine C1 and C2 to form cluster C = 500 + C5. Add C1, C2, and C to an ordered list of cell combinations, the clustering history record (CHR). Update the net lists to reflect this combination. Go to 2.

8. If any combinations were performed on the last pass through steps 2-7, go to step 2 restarting at the beginning of the net list.

9. If more than one cluster remains, combine clusters in pairwise fashion until only one cluster remains. As each new cluster is formed, note the formation in the CHR. When only one cluster remains, stop.

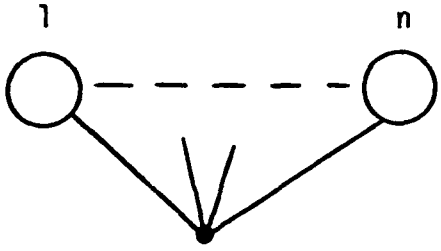
CLUSTERING VALUE COMPUTATION

A number of methods for computing and checking the clustering values against the threshold (steps 5 and 6 of the clustering procedure) are conceivable. The procedure used here considers effects of both direct and indirect connections between the cluster pair of interest. Also considered are effects of connections from the pair that may cause more profitable clustering of one element of the pair with some other cluster.

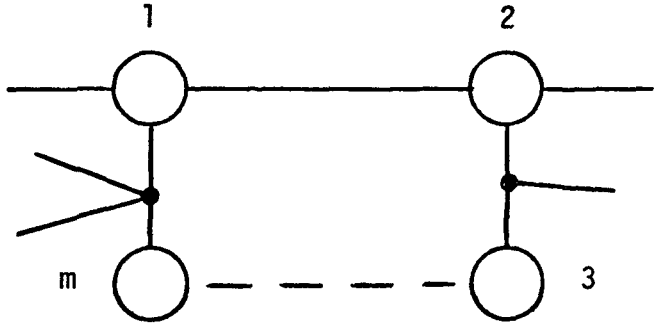
Prior to the description of the cluster value computation technique, the following terms will be defined:

n-net	- A net which interconnects exactly n clusters (Figure 6.a).
m-cycle	- A cycle of interconnections on exactly m clusters (Figure 6.b).
simple m-cycle	- An m-cycle in which each interconnection is a 2-net (Figure 6.c).
complex m-cycle	- An m-cycle which is not a simple m-cycle.

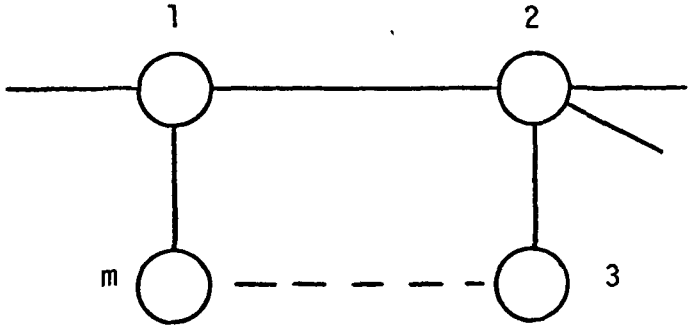
The cluster value computation procedure may now be described for two connected clusters, C1 and C2:



a) n-net



b) m-cycle



c) simple m-cycle

Figure 6. Terminology

1. Set Z3 = (# 2-nets incident to both C1 and C2), Z1 = (# nets incident to C1, but not C2), and Z2 = (# nets incident to C2, but not C1).

2. If Z3 - Z1 >= Z7 or Z3 - Z2 >= Z7 combine C1 and C2. Otherwise, proceed to step 3.

3. Set Z5 = (# 3-nets incident to both C1 and C2).

4. Set Z6 = Z3 - ((Z1 + Z2 - Z5) / 2) + .5. If Z6 >= Z7 combine C1 and C2. Otherwise, proceed to step 5.

5. Set Z8 = (# complex 3-cycles containing both C1 and C2). Set Z9 = (# simple 3-cycles containing both C1 and C2).

6. If Z6 + (Z8 / 2) + Z9 >= Z7 combine C1 and C2. Otherwise, C1 and C2 should not be combined.

Figure 7 shows several example network segments and Table 1 shows connection values for various cell pairs within the segments. Also shown in Table 1 is the maximum threshold value under which each cell pair will combine. Examination of this table shows that a higher maximum threshold for combination of a cell pair corresponds to a higher degree of 'connectedness' between the cells with respect to connections to other cells. A clustering for the network of Figure 4 is shown in Figure 8.

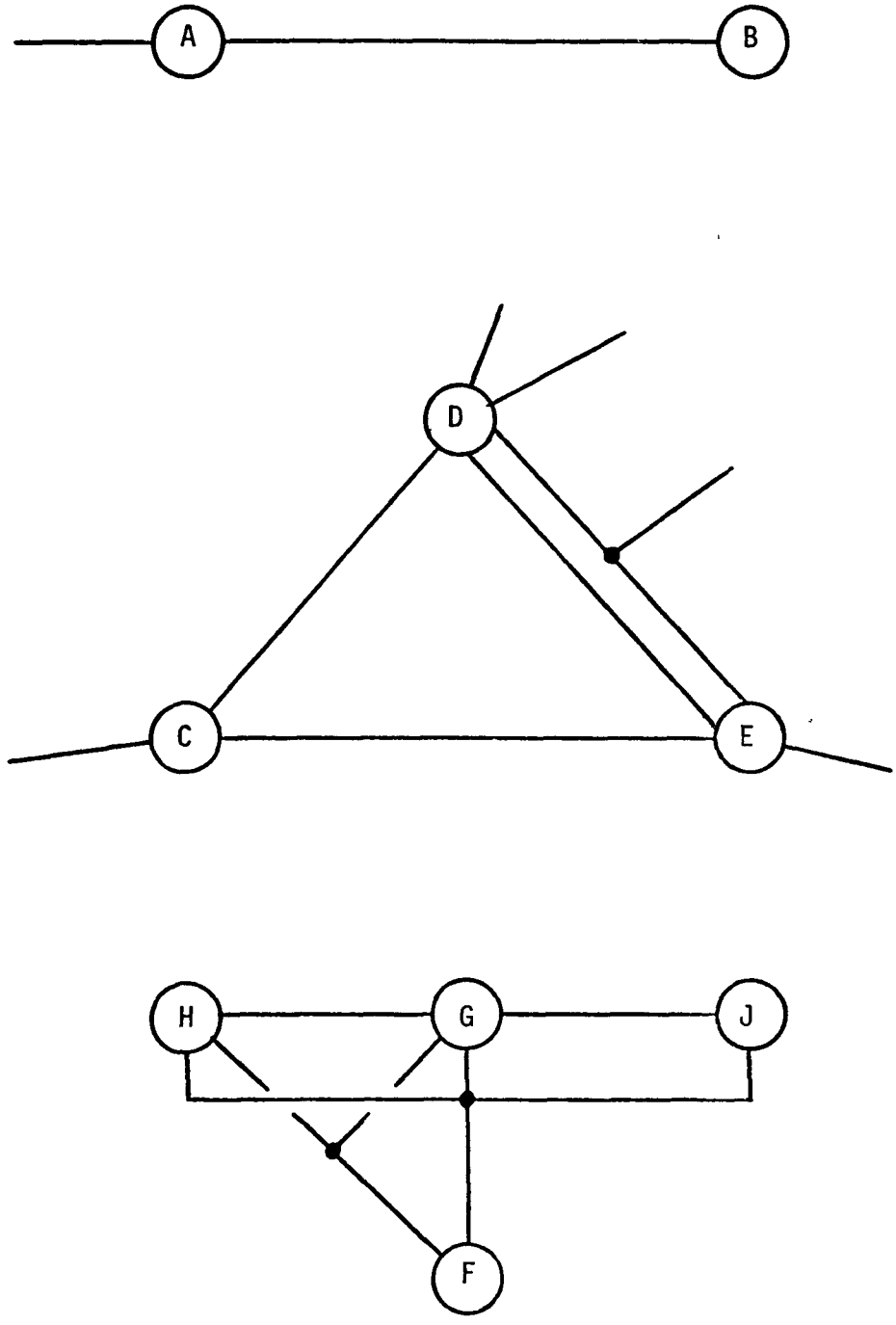


Figure 7. Typical Network Segments

TABLE 1. CELL COMBINATION EXAMPLES

CELLS		Z1	Z2	Z3	Z5	Z8	Z9	Z6	MAX Z7 FOR
C1	C2								COMBINATION
A	B	1	0	1	0	0	0	1	1
C	D	2	4	1	0	1	1	-1.5	0
C	E	2	3	1	0	1	1	-1	.5
D	E	3	2	1	1	1	1	-.5	1
F	G	0	2	0	1	2	0	0	1
F	H	0	1	0	1	2	0	.5	1.5
F	J	1	1	0	0	2	0	-.5	.5
G	H	1	0	1	1	2	0	1.5	2.5
G	J	2	0	1	0	2	0	.5	1.5
J	H	1	2	0	0	2	0	-1	0

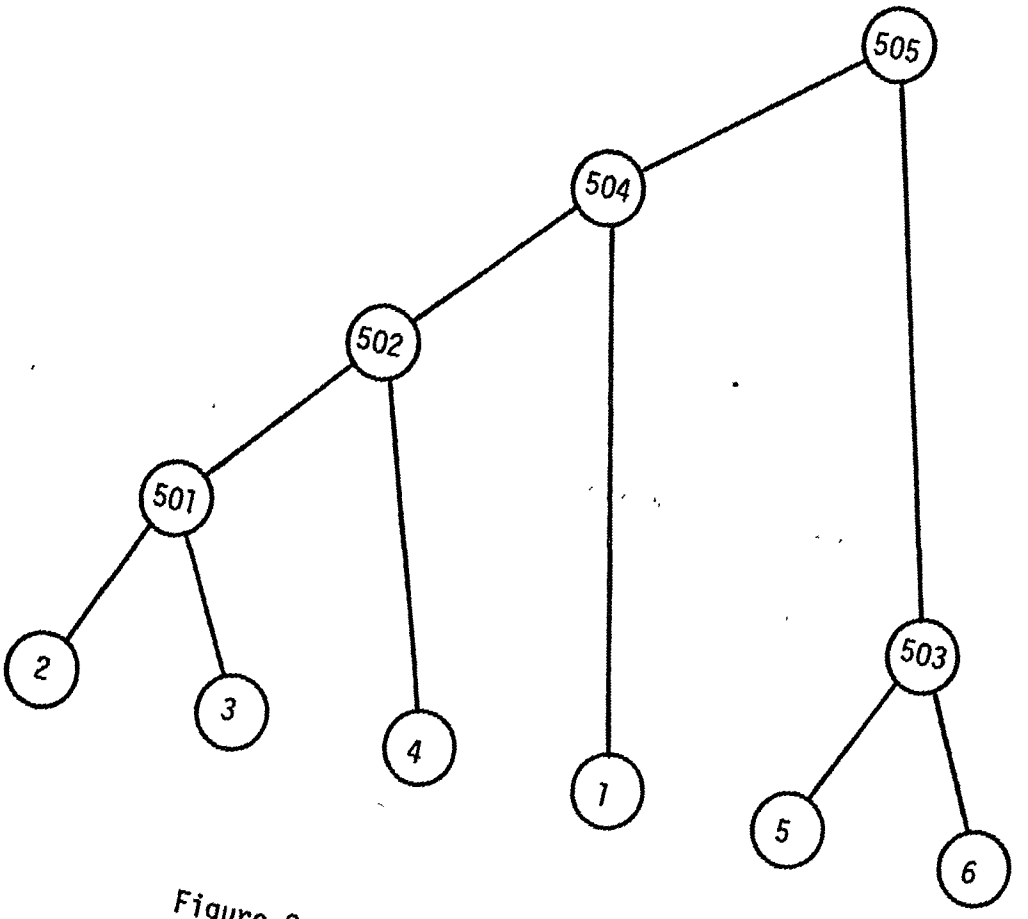


Figure 8. Clusters Formed for Network of Figure 4 (ALL = 81, WMAX = 20, Z7 = .5)

LINEAR CELL ORDER FORMATION

Following execution of the clustering task, the CHR is used to form a linear placement of the network cells in which the linear distance between connected cells is minimized. The procedure used for this placement is similar to the top-down linear placement technique described in [3].

The concept underlying this technique is that cells which were combined early in the clustering process are more strongly connected to each other than to other cells. These cells, then, should be made adjacent in the linear placement.

The linear ordering procedure will now be described.

1. From the CHR, obtain the highest-numbered cluster formed (the cluster containing the entire network). Split this cluster into its two constituents (the cluster pair used to form the cluster). Place these constituents in an arbitrary order in the linear placement.

2. Identify the highest-numbered cluster, CM, in the linear placement (if no clusters remain, stop). From the CHR, identify the constituents of CM, CA and CB.

3. We desire to replace CM with CA and CB in the placement. There are two possible orientations for these constituents. To identify the optimal orientation, consider the linear placement as

   NL CM NR

   where NL and NR are all the other clusters in the placement on the left and right side of CM, respectively. Then compute

   L(A) = # nets from CA to NL,

R(A) = # nets from CA to NR,

L(B) = # nets from CB to NL, and

R(B) = # nets from CB to NR.

4. Compute

$$X(A) = L(A) - R(A)$$

$$X(B) = L(B) - R(B).$$

If X(A) > X(B), CA is more strongly connected to NL and CB to NR, and the linear order should be

NL CA CB NR

otherwise, the order should be

NL CB CA NR.

5. Go to 2.

An example of this procedure is shown in Table 2.

PLACEMENT PROCEDURE

Following formation of the linear cell ordering, the linear placement must be mapped onto a STAR. The hypothesis underlying the placement procedure presented here is:

1. The linear placement constructed represents a near-optimal (linear) placement with regard to connection length. The two-dimensional placement, then, should preserve or reduce the intercell distances wherever possible. In particular, adjacent cells in the linear order should also be made adjacent on the STAR.

2. From the standpoint of the circuit designer the primary objective of the placement procedure is to fit the finite-size cells onto the STAR.

TABLE 2. FORMATION OF LINEAR PLACEMENT

FOR NETWORK OF FIGURE 4

PASS	CM	CA	CB	L(A)	L(B)	R(A)	R(B)	ORDER
0	-	-	-	-	-	-	-	503,504
1	504	502	1	1	0	0	0	503,502,1
2	503	6	5	0	0	1	1	5,6,502,1
3	502	501	4	1	1	1	0	5,6,4,501,1
4	501	3	2	1	1	1	1	5,6,4,2,3,1

The placement procedure, then, should be designed to achieve a balance of these two objectives.

BASIC PLACEMENT PROCEDURE

The placement method selected involves 'folding' the linear placement (with pads deleted) onto the STAR. The folding process may be accomplished by placing the cells from the linear ordering along a row in one direction and in the opposite direction on the next row as indicated in Figure 9.a. This organization, however, forces the majority of nets to pass through the rows horizontally. Since the horizontal global paths through the cells are limited, this procedure may result in overcrowding of net paths with resulting difficulty of routing.

An improved method of performing the folding operation forms double rows of cells from the linear ordering as illustrated in Figure 9.b. This organization allows some spreading of the nets both in the horizontal and vertical directions, resulting in reduced horizontal global path utilization. Almost all adjacencies from the linear placement are preserved by this method and most cells also become adjacent to the cells that they were previously one cell away from.

1	2	3	4
8	7	6	5
9	10	11	12
16	15	14	13

a) Simple Folding Technique

1	4	5	8
2	3	6	7
16	13	12	9
15	14	11	10

b) Improved Folding Techinique

Figure 9. Folding Techniques
(Linear Order = 1,2,---,16,
all widths = 1)

A procedure for performing this folding operation will now be presented. For convenience, the following notation is defined:

LMRn - Left-most unfilled region on row n

RMRn - Right-most unfilled region on row n

L(I) - Ith cell in linear order.

The placement procedure is:

1. Place L(1) at LMR1. Place L(2) at LMR2. LTOR=1 (1 for left-to-right placement, Ø for right-to-left), I=3 (linear list pointer), J=1 (J, J+1 are cell rows forming current double row), FLAG=Ø.

2. If all cells are placed, denote success and stop.

3. If there is more unfilled space on row J than on row J+1, K = J and L = J+1. If there is less space, K = J+1 and L = J. If the space is equal, keep the previous K and L.

4. If LTOR=Ø go to 6.

5. If L(I) will fit at LMRK, place it there, I = I+1, go to 2. Otherwise, go to 7.

6. If L(I) will fit at RMRK, place it there, I = I+1, go to 2.

7. If FLAG=Ø swap L and K, FLAG=1, go to 4.

8. LTOR = LTOR-1 (change placement direction for next double row), J = J+2, K = J, L = J+1, FLAG=Ø.

9. If J < # cell rows in array, go to 3. Otherwise, denote failure and stop.

An example placement obtained by use of this procedure is shown in Figure 10.

CELL SIZE CONSIDERATIONS

Experimental results show that sparsely populated (i.e. large ratio of array area to total cell area) STARs are easily laid out using this folding technique. As total cell area approaches array area, however, this procedure may result in unplaced cells after all appropriately-sized regions of the array are filled. A modification to the procedure is necessary to handle this situation.

The following points were considered in the development of this modification:

1. As shown in Figure 11, the previous procedure may succeed in placement of a network, yet fail for the same network with a different linear ordering. The modification, then, can include reordering of the linear placement followed by an attempted replacement on the STAR.

2. Since a balance between the placements specified by the linear ordering and the size restrictions of the array is desired, the modification should not change the linear placement to a degree greater than required.

3. The folding procedure outlined begins a new double row of cells whenever adding the next cell of the linear

LINEAR ORDER- 5 6 4 2 3

5	5	5	4	4	4	4	2	2
6	6	6	6	6	6	6	-	-
-	3	3	3	3	3	3	3	3

Figure 10. Placement for Network of Figure 4

LINEAR ORDER 1 2 3

1	1	3
2	2	2

a) Complete Placement

LINEAR ORDER 1 3 2

1	1	-
3	-	-

b) Incomplete Placement

Figure 11. Example of Placement Failure

order to the current double row would exceed the array width. The 'holes' remaining may be large enough to contain some other cell of the network. A modification of the procedure to fill these holes may result in complete placement.

MODIFIED PLACEMENT PROCEDURE

Based on these considerations, the modification to the basic procedure consists of the addition of two processes. The first of these, the rotation process, takes a linear ordering with form:

C1,C2,::::,CI,CJ,::::CN

and produces the new linear ordering:

CJ,::::,CN,C1,C2,::::,CI.

To reduce the effects of this reordering, a list of the number of nets crossing each cell boundary in the linear ordering is formed prior to execution of the placement routine. The cell boundary selected for performing a rotation is the boundary which is crossed by the minimum number of nets for all previously untried boundaries. In this way, the interconnecting nets disrupted by the rotation operation is held to a minimum.

The second process, the lookback process, consists of

the scanning of a limited number of preceding array rows for holes large enough to contain the current cell to be placed. If one is found, the hole is filled. Since this process may place a cell several rows from the location specified by the linear ordering, the lookback process is invoked only after failure of the rotation process.

The modified folding procedure is:

1. LBK=Ø (# rows for lookback process).

1a. Place L(1) at LMR1. Place L(2) at LMR2. LTOR=1 (1 for left-to-right placement, Ø for right-to-left), I=3 (linear list pointer), J=1 (J, J+1 are cell rows forming current double row), FLAG=Ø.

2. If all cells are placed, denote success and stop.

3. If there is more unfilled space on row J than on row J+1, K = J and L = J+1. Otherwise, K = J+1 and L = J.

3a. Scan rows J - LBK through J - 1. If L(I) will fit on one of these rows, place it there, I = I+1, go to 2.

4. If LTOR=Ø go to 6.

5. If L(I) will fit at LMRK, place it there, I = I+1, go to 2. Otherwise, go to 7.

6. If L(I) will fit at RMRK, place it there, I = I+1, go to 2.

7. If FLAG=Ø swap L and K, FLAG=1, go to 4.

8. LTOR = LTOR-1 (change placement direction for next double row), J = J+2, K = J, L = J+1, FLAG=Ø.

9. If J < # cell rows in array, go to 3.

10. If any rotations have not been tried, rotate original linear placement using minimum boundary crossing, go to 1a.

11. LBK = LBK + 1. If LBK > (# rows in array - 1), denote failure and stop. Otherwise, reset rotation trial

history (no rotations tried) and go to 1a.

The rotation process can be illustrated by considering a network with configuration identical to the network of Figure 4, but with cell widths as follows:

CELL #	WIDTH
1	0 (pad)
2	4
3	2
4	7
5	8
6	3

The clustering and linear ordering tasks for this altered network produce the linear order

5 6 4 2 3 1

as before. The boundary crossing data (neglecting pads) for this order is:

BOUNDARY LEFT CELL	BOUNDARY RIGHT CELL	NETS CROSSING BOUNDARY	RANK IN CROSSING LIST
5	6	4,5	2
6	4	4	1
4	2	3,4	3
2	3	1,2,3	4

Due to the altered cell widths, the original linear order cannot be placed on a 3-row by 9-column STAR. The first rotation of the order would be performed at the boundary between cells 6 and 4 to produce the order

4 2 3 5 6.

This ordering also cannot be placed on the STAR.

The second rotation would then be performed on the original order at the boundary between cells 5 and 6 to produce:

6 4 2 3 5

which can be placed as shown in Figure 12.

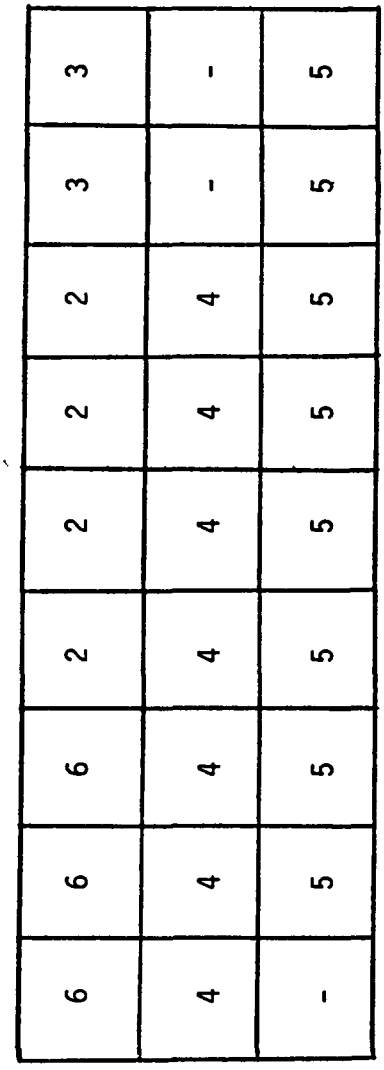


Figure 12. Placement of Network Requiring Rotation

III. PROGRAM IMPLEMENTATION

The procedures described in the preceding sections have been implemented as a series of programs in the BASIC-PLUS language on a PDP 11/40 time sharing system (RSTS/E v06C operating system). The high-level program and database flow is shown in Figure 13. A listing of each program is shown in Appendix A. The functional organization of each of the programs will be described in this section.

NETWORK INPUT PROGRAM (NETIN)

The network description task is handled by the network input program, NETIN. Four network description subtasks are performed by this program:

1. A name is assigned by the user to the network. This name is used for output reference and for database handling. In following sections, the network name will be referred to as 'netname'.

2. The list of cells in each net of the network is entered by the user and the lists are stored by NETIN in the file netname.NCL.

3. The width (in transistors) of each cell in the network is defined by the user and stored in the file netname.WID. Two options for cell width definition are provided:

   a) Assignment of default cell widths. This option causes all network cells to be defined as one transistor in width.

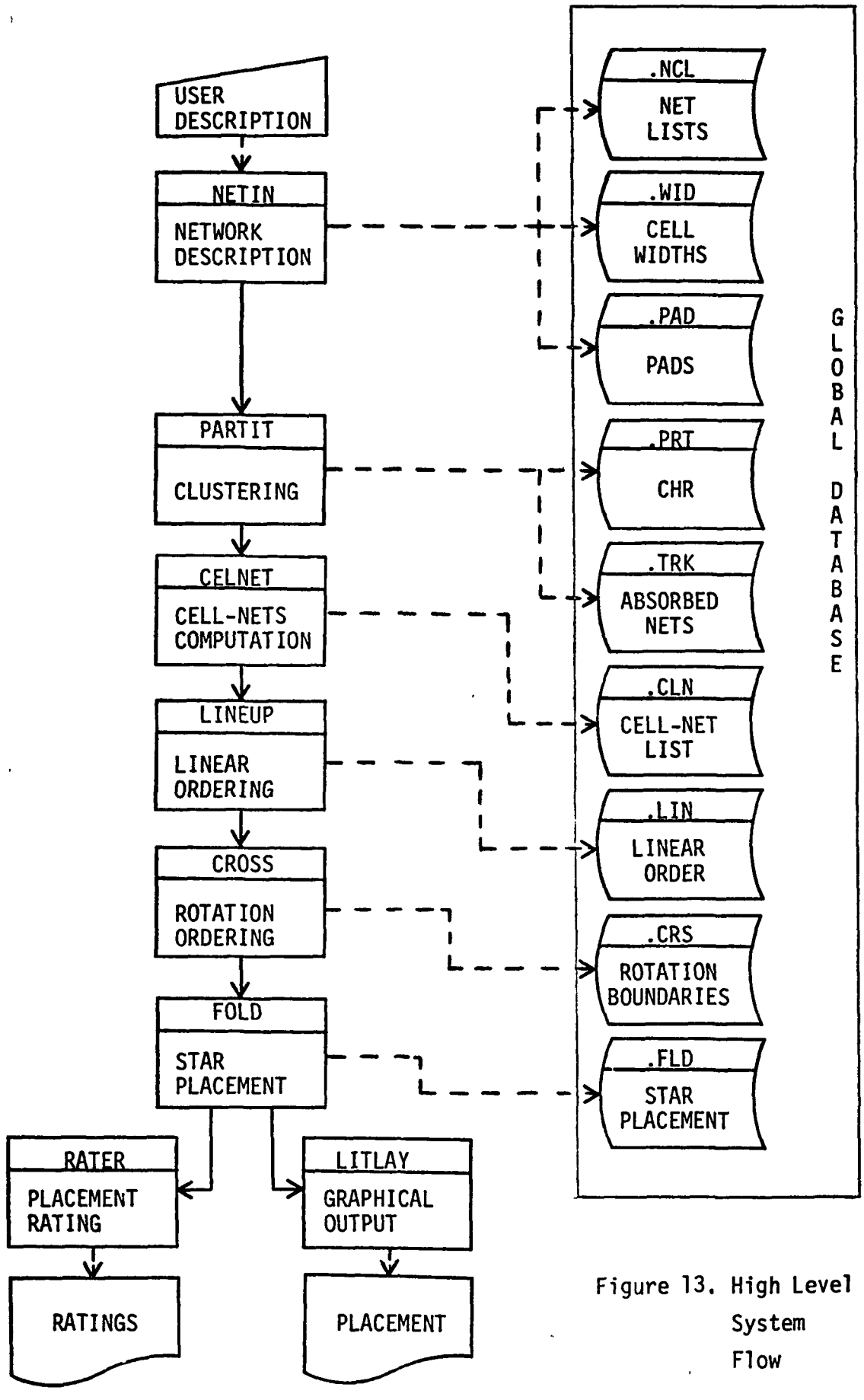


Figure 13. High Level System Flow

b) Exact specification of cell widths by the user. When this option is selected, any unspecified cell widths are set to the default width.

4. The pads of the network are defined by entering the user-assigned cell number used in the net lists. The width of each pad is set to 0. The pad list is stored by NETIN in the file netname.PAD.

An example NETIN execution is shown in Figure 14. The structure of the files netname.NCL, netname.WID, and netname.PAD is shown in Figure 15.

NETWORK CELL CLUSTERING (PARTIT)

The network cell clustering task is implemented by PARTIT. The logical flow of this program is as shown in the cell clustering procedure in a preceding section.

PARTIT obtains the network description data from the files netname.NCL and netname.WID (since pad identification is not necessary for the clustering procedure, the file netname.PAD is not used by PARTIT). Data entered by the user consists of:

1. netname,
2. value of WMAX (maximum partition size), and
3. value of Z7 (clustering value threshold).

Two databases are constructed by PARTIT. The first of these, netname.PRT, contains the clustering history record

```
NET NAME? XMPL


NET 1
   #CELLS (Ø TO STOP)? 3
? 1,2,3
NET 2
   #CELLS (Ø TO STOP)? 2
? 2,3
NET 3
   #CELLS (Ø TO STOP)? 2
? 3,4
NET 4
   #CELLS (Ø TO STOP)? 4
? 2,4,5,6
NET 5
   #CELLS (Ø TO STOP)? 2
? 5,6
NET 6
   #CELLS (Ø TO STOP)? Ø


DEFAULT WIDTHS (1)? NO

CELL #, WIDTH (Ø,Ø TO STOP)
? 2,2
? 3,8
? 4,4
? 5,3
? 6,7
? Ø,Ø

PAD (Ø TO STOP)
? 1
? Ø

Ready
```

Figure 14. Example NETIN Execution

netname.NCL

```
A1, A2, A3, 0, B1, B2, 0, --- , Q1, Q2, 0, -1
 cells in      cells in                    end of file
  net 1         net 2
```

netname.WID

```
W1, W2, W3, --- , WI, ---, WN
                  Width of
                   cell I
```

netname.PAD

```
P1, P2, ---, PI, ---, PM
             # of Ith
               pad
```

Figure 15. Structure of .NCL, .WID, .PAD Files

(CHR). The second database, netname.TRK, contains, for each net in the network, the number of the smallest cluster which completely contains the net. This information is used in later programs.

The data output to the user by PARTIT includes the net lists of the network, a record of cell combination, net lists and cluster sizes prior to the final step of the procedure (collapse), and the record of the collapsing step.

An example execution of PARTIT is shown in Figure 16. Organization of the files netname.PRT and netname.TRK is shown in Figure 17.

LINEAR PLACEMENT FORMATION (CELNET-LINEUP)

Two programs are used to implement the linear ordering task. The first of these, CELNET, uses the files netname.NCL, netname.PRT, and netname.TRK to produce netname.CLN. This database contains, for each cell in the network and for each cluster formed by PARTIT, a list of the nets that the cell or cluster is in. The structure of this file is shown in Figure 18.a.

The second program, LINEUP, uses netname.CLN and netname.PRT to perform the linear ordering procedure described in a previous section. The database produced,

```
NET NAME? XMPL
MAX CLUSTER SIZE (WMAX)? 20
CLUSTERING THRESHOLD (Z7)? .5
OUTPUT TO? KB:

********** INPUT DATA **********

NETNAME - XMPL      WMAX = 20          Z7 = .5

NET # 1          1  2  3

NET # 2          2  3

NET # 3          3  4

NET # 4          2  4  5  6

NET # 5          5  6

********** INITIAL CLUSTERING STEP **********

CELLS CONSIDERED          CLUSTER          CLUSTER
FOR COMBINATION           WIDTH            FORMED

 1  &  2
 1  &  3
 2  &  3                   10               501

 4  &  501                 14               502

 5  &  6                   10               503

 1  &  502

********** FINAL CLUSTERING STEP **********

 1  &  502                 14               504

 503  &  504               24               505

********** CLUSTER COMPOSITION **********

CLUSTER-- COMPOSED OF CELLS
 501       2  3
 502       4  2  3
 503       5  6
 504       1  4  2  3
 505       5  1  6  4  2  3
```

Figure 16. Example PARTIT Execution

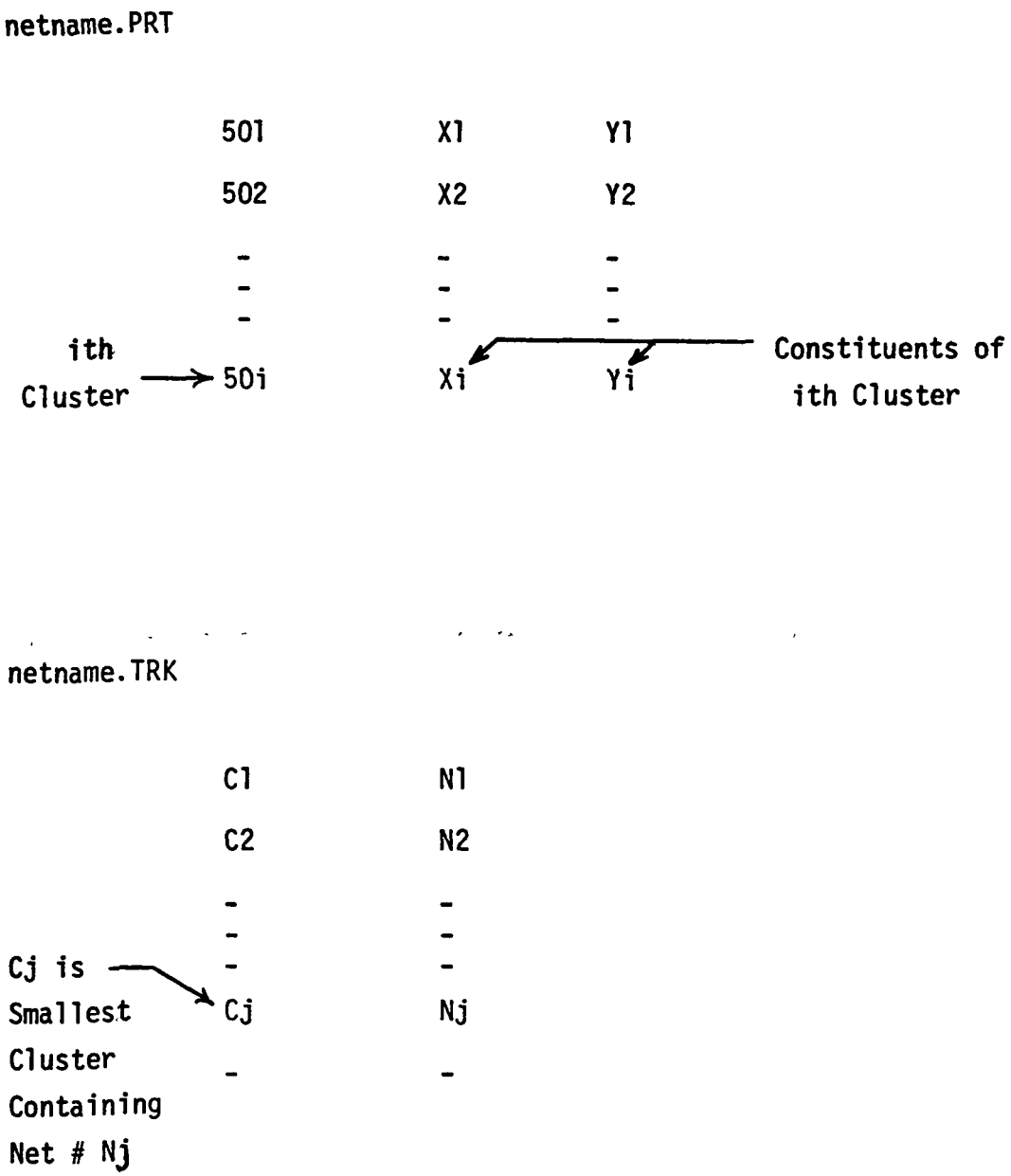


Figure 17. Structure of .PRT, .TRK Files

1.N1, 1.N2, ---, 1.NK, 2.NL, ---, J.NI, ---, 501.NP, ---

Cell #1 in Nets
N1, N2, ---, NK

a) netname.CLN

C1, C2, C3, ---, CI, ---, CN

Ith Cell in
Linear Order

b) netname.LIN

Figure 18. Structure of .CLN, .LIN Files

netname.LIN, contains the linear ordering for the network. The organization of this file is shown in Figure 18.b.

STAR PLACEMENT FORMATION (CROSS-FOLD)

The placement of the linear cell ordering on a STAR is accomplished by two programs. One of these, CROSS, forms the database netname.CRS which contains the linear cell boundary crossing data required for the rotation process in the placement procedure.

The second program, FOLD, executes the modified placement procedure previously described. User inputs to this program consist of the netname and the desired STAR size (cell rows by transistor columns). The output of this program consists of a history of lookback distance increase, the final (rotated) linear order, notification of placement success or failure, and the arrangement of the cells on the STAR. The placement is also output to form the file netname.FLD.

An example of the execution of FOLD is shown in Figure 19.

```
CHIP DIMS(R,C)? 3,9
NET NAME? XMPL
OUTPUT TO? KB:


INITIAL LOOKBACK DISTANCE IS 0
ALL CELLS PLACED AFTER ROTATION #  0

FINAL ROTATED LINEAR ORDER
 5
 6
 4
 2
 3




 5    5    5    4    4    4    4    2    2

 6    6    6    6    6    6    6    0    0

 0    3    3    3    3    3    3    3    3

Ready
```

Figure 19. Example FOLD Execution

UTILITY PROGRAMS (RATER, LITLAY)

To aid in the use and evaluation of the STAR placements generated, two utility programs have been developed. The first of these, RATER, forms a percentage rating of a cell placement (neglecting pads) based on the number of nets which could potentially be routed as straight connections. The procedure used computes, for each net, whether

1. All the cells of the net occupy a single cell row on the STAR,
2. There is at least one transistor column which passes through all the cells of the net, or
3. Neither 1 nor 2 occurs.

In case 3, there is no possibility of straight routing. If case 1 or 2 occurs, there is at least the potential for straight routing and the net is counted as straight.

The output of RATER consists of a list of net sizes (neglecting pads), the number of nets of each size, the total number of circuit nets, and the overall percentage of nets that are straight. An example of this output is shown in Figure 20.

The second utility program, LITLAY, forms a line-printer output of the cell placement as shown in Figure 21. In the output of this program, a cell is shown as follows:

```
NET NAME? XMPL

NET SIZE              NUMBER NETS              % STRAIGHT
 2                     4                        100

 4                     1                        0

TOTAL # NETS ANALYSED 5

        # STRAIGHT 4

        % STRAIGHT 80

Ready
```

Figure 20. Example RATER Execution

```
NET NAME? XMPL
PAPER WIDTH? 50
OUTPUT TO <KB:>? KB:
HEADER <NET NAME>? EXAMPLE LITLAY EXECUTION


EXAMPLE LITLAY EXECUTION


##################################################
#5-4 :5    :     #4-3 :4    :     :     #2-1 :2 4 #
# -  :     :     # -  :     :     :     # -  :     #
# -  :     :     # -  :     :     :     # -  :     #
##################################################
#6-4 :5    :     :     :     :     :     #0-  #0-  #
# -  :     :     :     :     :     :     # -  # -  #
# -  :     :     :     :     :     :     # -  # -  #
##################################################
#0-  #3-1 :2 3 :     :     :     :     :     :     #
# -  # -  :     :     :     :     :     :     :     #
# -  # -  :     :     :     :     :     :     :     #
##################################################


Ready
```

Figure 21. Example LITLAY Execution

```
# # # # # # # # #
# X - Y :  Z     #
# X - Y :  Z     #
# X - Y :  Z     #
# # # # # # # # #
```

where '#' represents the cell boundary, ':' indicates a transistor column boundary internal to the cell, 'XXX' is the cell number and 'YYY' and 'ZZZ' are the numbers of the nets that the cell is in.

Since the connection points of all nets are indicated in the LITLAY output, hand routing of the placed network can be easily accomplished.

IV. SYSTEM PERFORMANCE

Several example networks have been used for checking the operation of the placement system. The characteristics of four of these networks are shown in Table 3. System performance for these networks will be described in this section. Placement quality and other method testing results will be described first, followed by program execution timing.

PLACEMENT QUALITY

The quality measures of the STAR placements generated by this system have been computed based on the results of RATER executions (i.e., on the percentage of potentially straight nets). The quality testing scheme used is:

1. For a test network, form a STAR placement using the placement system and find the overall straight-net rating, T.

2. Perform a large number of random placements for the network (for the same STAR size) and save the rating of each.

3. Use the data from steps 1 and 2 to form an approximate normalized Gaussian distribution of placement ratings, G.

TABLE 3. TEST NETWORK PARAMETERS

NETWORK NAME	GATES	PADS	NETS	CELL SIZE RANGE	COMMENTS
XMPL	4	1	5	2 - 8	SHOWN IN FIG. 4
NAS101	25	14	35	4 - 16	MSFC's CO-101
NAS15	95	15	104	4 - 16	MSFC's CO-15
SAUCKT	26	5	33	1	CIRCUIT FROM [3]

4. Compute the placement quality, Q, by integrating G from its lower limit to the image of T. Then Q is the probability that the rating of any placement of the network is less than T.

The results of the placement quality testing procedure for the NAS101 and NAS15 networks are summarized in Table 4. These results indicate that over 99% of the possible placements for these networks can be expected to have lower straight net ratings than the system-generated placements. Tests performed on the XMPL and SAUCKT networks indicate similar high placement qualities.

Example system execution for the XMPL network has been shown in previous sections. System executions for the NAS101, NAS15, and SAUCKT networks are shown in Appendix B.

METHOD TESTING

Several other checks have been performed on the methods outlined. The objectives and results of these tests are outlined below.

1. The clustering method attempts to combine cells in the order in which they appear in the net list and not in the order of 'next-most clusterable cells'. To determine the effects of this procedure, a number of re-orderings of the net lists for the test networks were performed followed by placement and rating for the resulting lists. While variations in the formed

TABLE 4. PLACEMENT QUALITY

NETWORK	T	RANDOM PLACEMENTS (300 RUNS)		Q
		MEAN	S.D.	
NAS101	65.6%	45.7%	7.16%	.9973
NAS15	65.3%	26.32%	8.029%	.999999

clusters were noted, very little effect on the final placement ratings was produced indicating satisfactory clustering performance. In addition, the clustering method used should require less execution time than other techniques since the clustering value for all connected pairs of network cells is not computed prior to each cell combination.

2. Since the combination of a pair of cells depends to a high degree on the clustering value threshold, Z7, a study of the effects of variation of this parameter was performed. Results of this study, while providing no immediate relationship between Z7 and placement rating, indicate that a clustering threshold of Ø to .5 produces the most consistently high ratings.

3. The quality of the linear placement formed in the procedure can best be measured by computing the total length of connection medium required to route all nets in a linear manner (wire units). While a complete study of wiring length has not been performed, several linear placements of the SAUCKT network have been analysed. These studies indicate that results within 2 percent of optimum, as reported for the procedures of [3], are typical. Both linear placement methods resulted in a wire unit measure of 97 for this network.

4. No guaranteed solution is provided by the final placement method. Thus, the performance of this technique is of great concern. Over 9ØØ trial executions of FOLD with various networks and STAR sizes have been performed. No FOLD failures (i.e. the network can be placed on the STAR, but FOLD performs incomplete placement) have been identified. For high STAR densities, however, straight net ratings may decrease markedly due to the increased dependence on the rotation and lookback processes. It is anticipated that FOLD failures, if they exist, will be found at these high STAR densities (> 95% of transistors used) and should not occur for low-to-moderate STAR densities.

EXECUTION TIMING

Due to the multi-user environment in which the system programs are implemented, consistent program execution times are difficult to obtain. However, typical times for network placement (excluding network entry) have been identified. These execution times range from less than 1 minute for the XMPL network to over 100 minutes for the NAS15 network under heavy system loading.

Analysis of the execution of program segments shows that the most time-consuming operations are:

1. Reorganization of the net lists following each cell combination in PARTIT,

2. Sorting of the cell-net lists in CELNET, and

3. Computation of the left and right net counts in LINEUP.

Results from [3] indicate that procedures roughly equivalent to PARTIT, CELNET, and LINEUP execute in time approximately linear with the number of network cells. It is anticipated that improvements to the system described here will provide similar execution time characteristics.

V. CONCLUSION

Techniques for the placement of digital logic cells on STARs have been presented. The methods described have been computer-implemented and typical results have been shown.

Experimental results show near-optimum STAR placements can be achieved with respect to network routability while requiring a minimum of processing capability.

A number of areas for future work are indicated. Among these are:

1. Optimization of the existing system to provide execution speed in direct proportion to tne number of network cells.

2. Implementation of a pseudo-router for STAR placements to provide more exact placement ratings based on network routability. This program would not attempt to route the nets, but to identify situations which might lead to routing difficulties and, based on this, to produce a placement rating.

3. Implementation of a placement improvement procedure to optimize a STAR placement with respect to the ratings obtained from the pseudo-routing.

4. Modification of FOLD to assure complete placement of networks and to provide notification (short of exhaustive testing) of unplaceable networks.

5. Use and generalization of the placement ratings to identify modifications to the system which would allow further performance improvement.

6. Development of a portable FORTRAN IV version of the placement system.

REFERENCES

[1] Franson, Paul, "Don't Overlook Semicustom IC's for Your Next Design Project," EDN, Feb.5, 1977, pp.72-76.

[2] Edge,T.M., "Semicustom and Custom Integrated Circuits and the Standard Transistor Array Radix (STAR)," Electronics and Control Lab., Geo.C. Marshall Space Flight Center, RTOP: 506-18-31.

[3] Schuler,D. and E.Ulrich, "Clustering and Linear Placement," Proc. ACM-IEEE Design Automation Workshop, June 1972, pp.50-56.

APPENDIX A

PROGRAM LISTINGS

A listing of each program described in Cnapter III is given in this section.

NETIN

```
1!
        ! NETIN---- TAKES NET, PAD, WIDTH INFORMATION FROM KB
        !           AND LOADS IT IN DATA BASES X.NCL (NETS),
        !           X.PAD (PADS), AND X.WID (WIDTHS) WHERE X IS THE
        !           NAME ASSIGNED TO THIS NETWORK
        !
2       ON ERROR GO TO 600
3       EXTEND
5       DIM I%(100%)
6!
        ! FETCH NAME FOR NETWORK
        !
10      INPUT "NET NAME";A$
15!
        ! SET UP FILE FOR NETS
        !
20      OPEN A$+".NCL" FOR OUTPUT AS FILE 1%
30      DIM #1%,INP%(5000%)
35!
        ! INPUT NETS AND LOAD THEM INTO FILE (MAT INP%)
        !
40      I%=1%:
        INPPT%=1%
50      PRINT "NET";I%
60      INPUT "   #CELLS (0 TO STOP)";C%
65      IF C%=0% THEN 200
70      MAT I%=ZER(C%)
80      MAT INPUT I%
90      FOR J%=1% TO C%
100             INP%(INPPT%)=I%(J%)
110             INPPT%=INPPT%+1%
120     NEXT J%
125!
        ! PUT 0 AT END OF NET
        !
130     INP%(INPPT%)=0%
140     INPPT%=INPPT%+1%
150     I%=I%+1%
160     GO TO 50
180!
        ! PUT -1 AT END OF NET FILE
        !
200     INP%(INPPT%)=-1%
210     CLOSE 1%
215!
        ! SET UP WIDTH FILE
        !
220     OPEN A$+".WID" FOR OUTPUT AS FILE 2%
230     DIM #2%,WID%(200%)
235!
```

NETIN

```
        ! INITIALLY SET UP FILE FOR ALL CELLS WIDTH 1
        !
240     MAT WID%=CON
245!
        ! IF WIDTH 1 DESIRED, DON'T GET WIDTHS
        !
250     INPUT "DEFAULT WIDTHS (1)";B$
260     IF LEFT(B$,1%)="Y" THEN 500
265!
        ! GET CELL WIDTHS ( ANY UNSPECIFIED ARE WIDTH 1 UNLESS PAD )
        !
270     PRINT "CELL #, WIDTH (0,0 TO STOP)"
280     INPUT A%,B%
290     IF A%=0% THEN 500
300     WID%(A%)=B%:
        GO TO 260
400!
        ! SET UP PAD FILE
        !
500     OPEN A$+".PAD" FOR OUTPUT AS FILE 3%
510     DIM #3%,PAD%(50%)
515     MAT PAD%=ZER:
        K%=1%
516!
        ! GET PAD #S
        !
520     PRINT "PAD (0 TO STOP)"
530     INPUT A%
540     IF A%=0% THEN 600
550     PAD%(K%)=A%
560     K%=K%+1%:
        WID%(A%)=0%:
        GO TO 530
599!
        ! ERROR HANDLER AND NORMAL EXIT
        !
600     CLOSE 1,2,3
700     END
```

```
PARTIT

1!
        ! PARTIT--- BASIC TRANSLATION OF FORTRAN PARTITIONER
        !           SENT FROM NASA
        !
10!
        ! VERSION 2
        !
30!
        ! MODIFICATION HISTORY
        !
40!
        ! 5-JAN-78        ADDED CONSTRUCTION OF X.TRK
        !                TO ALLOW DELETION OF ABSORBED
        !                NETS IN CELNET
        !
1000    EXTEND
10000   ON ERROR GO TO 20000
10030   !IMPLICIT INTEGER(A-Y)
10040   DIM TEMP%(101%):
        DIM BANK%(100%):
        DIM FINET%(30%):
        DIM REG%(100%)
10050   DIM NC%(200%):
        DIM PC%(200%):
        DIM BC%(1000%):
        DIM CELL%(200%)
10060   DIM NN%(200%):
        DIM PN%(200%):
        DIM BN%(1000%)
10065   DIM WHI%(200%):
        DIM CIDHI%(200%):
        DIM CIDLO%(200%)
10070   DIM WMI%(200%):
        DIM CIDMI%(200%)
10075   DIM T2%(200%)
10076   DIM ZQ2(200%)
10077   DIM ZQ1(200%)
10090   OODBG%=1%
10100   FLG1%=0%
10110   C5%=0%
10120   ALL%=81%
10130   ! WMAX IS MAX # CELLS IN A PARTITION
10140   INPUT "MAX PARTITION WIDTH";WMAX%
10150   ! LARGER Z7 IS, LESS LIKELY CELLS TO BE COMBINED
10160   INPUT "CLUSTERING THRESHOLD";Z7
10180   IX%=1%
10181!
        ! GET NETWORK NAME
        !
10182   INPUT "NET NAME";NNME$
```

PARTIT

```
10183!
        ! FETCH NET, WIDTH DATA
        !
10184   OPEN NNME$+".NCL" FOR INPUT AS FILE 1%
10185   DIM #1%,INP%(5000%)
10186   OPEN NNME$+".WID" FOR INPUT AS FILE 5%
10187   DIM #5%,WID%(200%)
10188   INPUT "OUTPUT TO";B$
10189   OPEN B$ FOR OUTPUT AS FILE 2%
10190   IF OODBG%=0% THEN 10230
10192   OPEN NNME$+".TRK" FOR OUTPUT AS FILE 7%
10194   DIM #7%,TRACE%(500%,2%)
10196   TRACK%=1%
10200   PRINT #2%, "INPUT DATA"
10210   PRINT #2%,"WMAX =";WMAX%:
        PRINT #2%,"Z7 =";Z7
10230   NC%(I%)=0% FOR I%=1% TO 200%:
        CIDLO%(I%)=0% FOR I%=1% TO 200%:
        CIDHI%(I%)=0% FOR I%=1% TO 200%:
        PC%(I%)=0% FOR I%=1% TO 200%:
        WHI%(I%)=0% FOR I%=1% TO 200%
10240   CIDMI%(I%)=0% FOR I%=1% TO 200%:
        WMI%(I%)=0% FOR I%=1% TO 200%.
10300   ! ZERO BC% ARRAY
10310   BC%(QQ%)=0% FOR QQ%=1% TO 1000%
10350   R%=0%
10360   PS%=1%
10380   ! ZERO TEMP ARRAY
10385   INPPT%=0%
10390   TEMP%(QQ%)=0% FOR QQ%=1% TO 101%
10430   ! READS NETS FROM UNIT 1,PLACES IN TEMP ARRY
10440   I%=1%
10445   INPPT%=INPPT%+1%
10450   IF INP%(INPPT%)=0% THEN 10460
10452   IF INP%(INPPT%)=-1% THEN 11010
10453   IF I%>100% THEN PRINT #2%,"ERROR - TOO MANY CELLS IN NET":
                GO TO 20000
10454   TEMP%(I%)=INP%(INPPT%)
10455   I%=I%+1%
10456   GO TO 10445
10460   IF OODBG%=0% THEN 10690
10465   PRINT #2%," ":
        PRINT #2%," ":
        PRINT #2%,"NET #";R%+1%;
10470   J%=1%
10480   PRINT #2%,TAB(15%);
10490   FOR J3%=0% TO 8%
10500     IF J%+J3%=I% THEN 10690
10510     PRINT #2%,TEMP%(J%+J3%);
10520   NEXT J3%
```

PARTIT

```
10525    PRINT #2%," "
10530    J%=J%+9%:
         GO TO 10480
10680    ! ARRANGES ELEMENTS IN EACH NET IN ORDER
10690    FOR J%=1% TO 100%
10700      N%=J%+1%
10705      IF TEMP%(J%)=0% THEN 10820
10710      FOR I%=N% TO 100%
10720            IF TEMP%(I%)=0% THEN 10780
10730            IF TEMP%(J%)<=TEMP%(I%) THEN 10770
10740            STOR%=TEMP%(J%)
10750            TEMP%(J%)=TEMP%(I%)
10760            TEMP%(I%)=STOR%
10770      NEXT I%
10780    NEXT J%
10800    ! DELETES THE DOUBLE ELEMENT IN EACH NET LINE AND
10810    ! STORES IN BANK ARRAY.
10820    N%=0%
10840    ! ZERO BANK ARRAY
10850    BANK%(QQ%)=0% FOR QQ%=1% TO 100%
10890    ! ARRANGE BANK SAME AS TEMP, BUT DELETE REPEATED ELEMENTS
10900    FOR I%=1% TO 101%
10910      IF TEMP%(I%)=0% THEN 10970
10920      IF TEMP%(I%)=TEMP%(I%+1%) THEN 10950
10930      N%=N%+1%
10940      BANK%(N%)=TEMP%(I%)
10950    NEXT I%
10970    R%=R%+1%
10980    ! FORM NC, PC, BC% ARRAYS
10990    GO SUB 11280
11000    GO TO 10390
11010    CLOSE 1%
11011    PRINT #2%," ":
         PRINT #2%," "
11015    PRINT #2%,"CELLS TO BE";TAB(30%);"COMBINED";TAB(50%);"NEW CELL"
11017    PRINT #2%,"COMBINED";TAB(30%);"CELL WIDTH"
11100    GO SUB 11550
11110    GO SUB 12760
11130    PRINT "GOOD RUN"
11140    PRINT #2%,"GOOD RUN"
11150    GO TO 20000
11160!
         !********* SUBROUTINE POINT1 *************
         ! PUTS NUMBER OF CELLS IN NC(R) - PUTS STARTING
         ! LOCATON IN PC & STORES CELL # IN BC
         !
11280    NC%(R%)=N%
11290    PC%(R%)=PS%
11320    BC%(PS%+(I%-1%))=BANK%(I%) FOR I%=1% TO NC%(R%)
11340    PS%=PS%+(NC%(R%))
```

PART1T

```
11350   RETURN
11360!
        ! ************ SUBROUTINE POINT2 ************
        ! MAKES ARRAYS CELL, NUMBER OF NETS (NN), POINTER
        ! FOR NETS (PN) & ARRAY CONTAINING NETS (BN)
        !
11550   IN%=1%
11570   ! ZERO NN, PN, BN ARRAYS
11590   CELL%(I%),NN%(I%),PN%(I%),BN%(I%)=0% FOR I%=1% TO 200%
11600   BN%(I%)=0% FOR I%=201% TO 1000%
11710   ! IF NC(IN) = 1 THEN NET IS GONE INTO PARTITION
11720   IN%=IN%+1% WHILE NC%(IN%)=1%
11760   IB%=1%
11765   GWC30%=PC%(IN%)
11770   ! IF PC%(IN)=0 THEN END OF FILE
11780   IF GWC30%=0% THEN 12100
11790   FOR IJ%=GWC30% TO GWC30%+NC%(IN%)-1%
11820     BANK%(IB%)=BC%(IJ%)
11830     IB%=IB%+1%
11840   NEXT IJ%
11850   BANK%(IB%)=0%
11860   ! ***** MAKES CELL ARRY AND NUMBER OF NETS LOCATED IN(NN)**
11870   FOR I%=1% TO 200%
11875     GWC20%=BANK%(I%)
11880     JJ%=1%
11890     IF GWC20%=0% THEN IN%=IN%+1%:
                GO TO 11720
11900     IF CELL%(JJ%)=0% THEN 11980
11910     IF CELL%(JJ%)<>GWC20% THEN JJ%=JJ%+1%:
                      GO TO 11900
11920     NN%(JJ%)=NN%(JJ%)+1%:
          GO TO 12020
11980     IF FLG1%<>1% THEN WHI%(JJ%)=GWC20%:
                    WMI%(JJ%)=WID%(GWC20%)
12000     CELL%(JJ%)=GWC20%
12010     NN%(JJ%)=1%
12020   NEXT I%
12030   PRINT #2%,"ERROR--0 NOT FOUND IN LAST EL OF NET ARRAY"
12050   GO TO 20000
12070   ! ******MAKES ARRYS PN(POINTER FOR NETS) AND BANK FOR NETS.
12100   FLG1%=1%:
        PN%(1%)=1%:
        PN%(I%)=PN%(I%-1%)+NN%(I%-1%) FOR I%=2% UNTIL CELL%(I%)=0%
12160   NET%=1%
12170   NET%=NET%+1%:
        NET%=NET%+1% WHILE NC%(NET%)=1%
12190   OO20%=PC%(NET%):
        IF OO20%=0% THEN 12430
12220   FOR I%=OO20% TO OO20%+NC%(NET%)-1%
12225     GWC21%=BC%(I%):
```

```
PARTIT

          IF GWC21%=0% THEN 12430
12240     J%=1%:
          J%=J%+1% UNTIL (CELL%(J%)=GWC21% OR J%=200%)
12250     IF J%=200% THEN PRINT #2%,"ERROR--CELL";
                  BC%(I%);"NOT IN CELL ARRAY":
                  GO TO 20000
12300     GWC29%=PN%(J%)+NN%(J%)-1%:
          K%=PN%(J%):
          K%=K%+1% UNTIL (BN%(K%)=0% OR K%=GWC29%)
12330     IF BN%(K%)<>0% THEN PRINT #2%,"ERROR--NO SPACE IN BN ARRAY":
                        GO TO 20000
12380     BN%(K%)=NET%
12390   NEXT I%
12420   GO TO 12170
12430   RETURN
12590!
        !*************** SUBROUTINE REORG ********************
        !
12760   SAVC5%=C5%
12770   FOR II%=1% TO 200%
12775     GWC10%=NC%(II%):
          GWC22%=PC%(II%)
12780     IF C5%>ALL% THEN 12800
12790     IF GWC10%>3% THEN 13100
12795     IF GWC10%=2% THEN 12830
12800     IF GWC10%=1% THEN 13100
12810     IF GWC10%=0% AND SAVC5%=C5% THEN 13160
12820     IF GWC10%=0% THEN 12760
12830     FOR JJ1%=GWC22% TO GWC22%+GWC10%-2%
12860           C1%=BC%(JJ1%)
12870           FOR JJ2%=JJ1%+1% TO GWC22%+GWC10%-1%
12900             C2%=BC%(JJ2%):
                  W1%,W2%,WB%=0%
12910             FLG9%=0%
12930             FOR I%=1% UNTIL FLG9%=2%
12935                   GWC11%=WHI%(I%)
12940                   IF GWC11%=C1% THEN W1%=WMI%(I%):
                                WB%=WB%+W1%:
                                FLG9%=FLG9%+1%
12950                   IF GWC11%=C2% THEN W2%=WMI%(I%):
                                WB%=WB%+W2%:
                                FLG9%=FLG9%+1%
12960                   IF WB%>WMAX% THEN 13080
12990             NEXT I%
13000             COMPL%=500%:
                  PRINT #2%," ":PRINT #2%,C1%;" & ";C2%;:
                  GO SUB 14070
13040             IF COMPL%=500% THEN 13080
13050             GO SUB 15090
13070             GO TO 13100
```

PARTIT

```
13080           NEXT JJ2%
13090     NEXT JJ1%
13100   NEXT II%
13130   !*** DETERMINES THE FINAL CELLS IN CID, AND WRITES TO IO 13
13160   PRINT #2%," "
13170   PRINT #2%,"CELL--COMPOSED OF CELLS"
13180   IF CIDHI%(1%)=0% THEN PRINT #2%,"NO CELLS COMBINED":
                    GO TO 13930
13190   FOR KI%=1% TO 200%
13195     IF CIDHI%(KI%)=0% THEN 13650
13200     X1%=CIDHI%(KI%):
          C2%=CIDLO%(KI%):
          C1%=CIDMI%(KI%)
13220     MAT REG%=ZER:
          REG%(1%)=X1%:
          REG%(2%)=C1%:
          REG%(3%)=C2%:
          FLG5%=0%
13280     FLG5%=0%
13290     FOR IZ%=2% TO 100%
13295           GWC23%=REG%(IZ%):
                IF GWC23%<500% THEN 13500
13300           IF GWC23%<500% THEN 13500
13310           FLG5%=1%
13320           FOR I%=1% TO 200%
13330             IF GWC23%=CIDHI%(I%) THEN 13380
13340           NEXT I%
13350           PRINT #2%,"ERROR 680"
13360           GO TO 20000
13380           X1%=CIDHI%(I%):
                C2%=CIDLO%(I%):
                C1%=CIDMI%(I%)
13400           REG%(IZ%)=C1%
13420           FOR IY%= IZ%+1% TO 100%
13430             IF REG%(IY%)=0% THEN REG%(IY%)=C2%:
                          GO TO 13290
13440           NEXT IY%
13450           PRINT #2%,"ERROR 640"
13460           GO TO 20000
13500           IF REG%(IZ%)=0% THEN 13550
13510     NEXT IZ%
13520     PRINT #2%,"ERROR 615"
13530     GO TO 20000
13550     IF FLG5%<>0% THEN 13260
13570     IF REG%(1%)=0% THEN 13600
13584           PRINT #2%, REG%(I%); FOR I%=1% UNTIL REG%(I%)=0%
13590     PRINT #2%," "
13600   NEXT KI%
13650   PRINT #2%," "
13655   PRINT #2%,"FINAL CIRCUIT NETS"
```

PARTIT

```
13656   PRINT #2%,"NET NO."
13670   FOR I%=1% UNTIL NC%(I%)=0%
13680     IK%=1%
13690     IF NC%(I%)=1% THEN 13820
13740     FOR J%=PC%(I%) TO PC%(I%)+NC%(I%)-1%
13770           FINET%(IK%)=BC%(J%)
13780           IK%=IK%+1%
13790     NEXT J%
13800     PRINT #2%," "
13802     PRINT #2%,I%,
13805     PRINT #2%, FINET%(G1%); FOR G1%=1% TO IK%-1%
13815     PRINT #2%," "
13820   NEXT I%
13830   PRINT #2%," "
13832   PRINT #2%,"CELL WIDTHS"
13850   FOR I%=1% TO 200%
13860     IF WHI%(I%)+WMI%(I%)=0% THEN 13930
13870     PRINT #2%, WHI%(I%);WMI%(I%)
13890   NEXT I%
13900   PRINT #2%,"ERROR IN WRITE OF WIDTH ARRAY"
13910   GO TO 20000
13930   GO SUB 16000
13931   OPEN NNME$+".PRT" FOR OUTPUT AS FILE 4%
13932   DIM #4%,CHI%(200%),CMI%(200%),CLO%(200%)
13933   MAT CHI%=CIDHI%:
       MAT CMI%=CIDMI%:
       MAT CLO%=CIDLO%
13934   CLOSE 4%
13935   !
13940   !
13950   GO TO 20000
13960   !
13970   !**********************SUBROUTINE JONES**********************
14070   Z1,Z2,Z3,Z5,Z6=0
14130   J1%=0%
14140   J2%=0%
14150   I%=1%
14160   I%=I%+1% UNTIL C1%=CELL%(I%)
14180   NNC1%=NN%(I%)
14185   GWC13%=PN%(I%)
14190   J%=1%
14200   J%=J%+1% UNTIL C2%=CELL%(J%)
14205   GWC14%=PN%(J%)
14220   NNC2%=NN%(J%)
14230   FOR I1%=0% TO NNC1%-1%
14235     GWC24%=BN%(GWC13%+I1%)
14250     FOR I2%=0% TO NNC2%-1%
14270           IF GWC24%=BN%(GWC14%+I2%) THEN 14350
14280     NEXT I2%
14290     J1%=J1%+1%
```

PARTIT

```
14300      Z1=Z1+1
14310      T1%(J1%)=GWC24%
14320      ZQ1(J1%)=0.5
14330      IF NC%(T1%(J1%))=2% THEN ZQ1(J1%)=1
14340      GO TO 14420
14350      GWC1%=NC%(GWC24%)
14355      IF GWC1%=2% THEN Z3=Z3+1
14360      IF GWC1%=3% THEN Z5=Z5+1
14420    NEXT I1%
14430    FOR I2%=0% TO NNC2%-1%
14435      GWC25%=BN%(GWC14%+I2%)
14450      FOR I1%=0% TO NNC1%-1%
14470            IF GWC25%=BN%(GWC13%+I1%) THEN 14540
14480      NEXT I1%
14490      J2%=J2%+1%
14500      Z2=Z2+1
14510      T2%(J2%)=BN%(PN%(J%)+I2%)
14520      ZQ2(J2%)=0.5
14530      IF NC%(T2%(J2%))=2% THEN ZQ2(J2%)=1
14540    NEXT I2%
14550    IF Z3>=(Z1+Z7) OR Z3>=(Z2+Z7) THEN 14880
14570    Z6=Z3-(Z1+Z2-Z5)/2+.5-Z7
14600    IF Z6>=0 THEN 14880
14610    FOR N1%=1% TO J1%
14620      FOR N2%=1% TO J2%
14630            IF ZQ2(N2%)=0 THEN 14850
14640            OO25%=PC%(T1%(N1%))
14650            OO12%=OO25%+NC%(T1%(N1%))-1%
14660            FOR L1%=OO25% TO OO12%
14670              IF L1%=C1% THEN 14840
14680              OO26%=PC%(T2%(N2%))
14690              OO13%=OO26%+NC%(T2%(N2%))-1%
14700              FOR L2%=OO26% TO OO13%
14710                    IF L2%=C2% THEN 14830
14720                    IF BC%(L1%)<>BC%(L2%) THEN 14830
14740                    IF ZQ1(N1%)=ZQ2(N2%) THEN Z6=Z6+ZQ1(N1%)
                         ELSE Z6=Z6+.5
14780                    IF Z6>=0 THEN 14880
14790                    ZQ1(N1%)=ZQ1(N1%)-0.5
14800                    ZQ2(N2%)=ZQ2(N2%)-0.5
14810                    IF ZQ1(N1%)=0 THEN 14860
                         ELSE 14850
14820                    GO TO 14850
14830              NEXT L2%
14840            NEXT L1%
14850      NEXT N2%
14860    NEXT N1%
14870    GO TO 14900
14880    C5%=C5%+1%
14890    COMPL%=COMPL%+C5%
```

PARTIT

```
14900   RETURN
14910!
        !************ SUBROUTINE REDO ****************
        ! IF COMP IS RETURNED OTHER THAN
        ! 500, NC & BC  ARRAYS ARE REDONE
        !
15090   FLGW%=0%
15100   FOR I%=1% TO 200%
15105     GWC26%=WHI%(I%)
15110     IF GWC26%=0% THEN
          PRINT #2%,"CELLS NOT FOUND IN REDO FOR WEIGHTS":
          GO TO 20000
15120     IF GWC26%=C1% THEN W1%=WMI%(I%):
                    GO TO 15200
15130     IF GWC26%=C2% THEN W2%=WMI%(I%)
                     ELSE 15220
15200     IF FLGW% THEN 15290
                  ELSE FLGW%=FLGW%+1%: POINT%=I%
15220   NEXT I%
15290   FOR I%=1% TO 200%
15295     GWC27%=WHI%(I%):
          IF GWC27%=0% THEN 15420
15300     IF GWC27%=C1% OR GWC27%=C2% THEN 15370
15330   NEXT I%
15340   PRINT #2% "ERROR IN WIDTH ARRAY, CELL C1 OR C2 NOT FOUND":
        GO TO 20000
15370   WB%=W1%+W2%:
        WHI%(I%)=COMPL%:
        WMI%(I%)=WB%
15400   ! COMBINES THE THREE CELLS INTO ONE (COMPL,C1,C2)
15420   CIDHI%(IX%)=COMPL%:
        CIDMI%(IX%)=C1%:
        CIDLO%(IX%)=C2%
15430   PRINT #2%,TAB(30%);WB%;TAB(50%);COMPL%
15470   IX%=IX%+1%
15480   FOR I%=1% TO 200%
15485     GWC28%=NC%(I%)
15490     IF GWC28%=1% THEN 15790
15500     IF GWC28%=0% THEN 15830
15510     INUM%=0%:
          OO27%=PC%(I%):
          OO14%=OO27%+GWC28%-1%
15540     FOR J%=OO27% TO OO14%
15550           IF BC%(J%)<>C1% AND BC%(J%)<>C2% THEN 15670
15580           IF INUM% THEN 15720
                   ELSE INUM%=INUM%+1%
15610           BC%(LL%)=BC%(LL%+1%) FOR LL%=J% TO OO14%-1%
15650           BC%(OO14%)=COMPL%
15660           GO TO 15550
15670     NEXT J%
```

PARTIT

```
15680      GO TO 15790
15690      ! THIS PART MOVES ALL BC% ARRY UP ONE POSITION IN THAT NET
15700      ! AND ASSIGNS -1-TO LAST POSITION.
15720      BC%(K%)=BC%(K%+1%) FOR K%=J% TO OO14%-1%
15730      BC%(OO14%)=-1%
15780      NC%(I%)=GWC28%-1%
15785      IF NC%(I%)=1% THEN TRACE%(TRACK%,1%)=C5%+500%:
                   TRACE%(TRACK%,2%)=I%:
                   TRACK%=TRACK%+1%
15787      IF NC%(I%)=1% THEN TRACE%(TRACK%,1%)=0%
15790    NEXT I%
15800    PRINT #2%,"ERROR IN REDO AFTER STATEMENT 235"
15820    GO TO 20030
15830    GO SUB 11550
15840    ! ***** RETURNES TO REORG
15850    RETURN
16000!
         !************ SUBROUTINE COLLAPSE **************
         !
16005    PRINT #2%,"COLLAPSE"
16010    C6%=C5%
16020    FOR II%=1% TO 200%
16030      IF NC%(II%)=1% THEN 16130
16040      IF NC%(II%)=0% AND C6%=C5% THEN 16140
16050      IF NC%(II%)=0% THEN 16010
16070      C1%=BC%(PC%(II%)):
           C2%=BC%(PC%(II%)+1%)
16090      PRINT #2%," ":
           PRINT #2%,C1%;" & ";C2%;
16100      C5%=C5%+1%:
           COMPL%=C5%+500%
16110      GO SUB 15090
16130    NEXT II%
16140    RETURN
20000    CLOSE 1,2,3,4
20005    CLOSE 7
20010    IF ERR<>0 THEN PRINT "ERROR #";ERR;"AT LINE";ERL
20020    NO EXTEND
32767    END
```

CELNET

```
1!
        ! CELNET--- SETS UP X.CLN DATA BASE WITH ENTRIES OF FORM
        !                                   PPP.QQQ
        !           WHERE PPP IS A CELL # AND QQQ IS A NET NUMBER
        !           ALL CELLS (INCLUDING THOSE GENERATED BY PARTIT)
        !           AND ALL NETS ARE TREATED
        !
20!
        ! VERSION 2
        !
30!
        ! MODIFICATION HISTORY
        !
40!
        ! 5-JAN-78              MODIFIED METHOD OF ADDING ENTRIES FOR
        !                       PARTIT-FORMED CELLS TO PREVENT INSER-
        !                       TING CELL-NETS THAT ARE COMPLETELY
        !                       ABSORBED.  USES X.TRK TO FIND ABSORBED
        !                       NETS.
1000    EXTEND
2000    DIM C1%(200%),C2%(200%),C3%(200%)
2010    DIM C(5000%)
2020    DIM N0%(200%)
10000   ON ERROR GO TO 20000
10004!
        ! FETCH NET FILE AND PARTITION FILE
        !
10005   INPUT "NET NAME";NNME$
10006   OPEN NNME$+".NCL" FOR INPUT AS FILE 2%
10010   OPEN NNME$+".PRT" FOR INPUT AS FILE 1%
10020   DIM #1%,CHI%(200%),CMI%(200%),CLO%(200%)
10030   DIM #2%, INP%(5000%)
10032   OPEN NNME$+".TRK" FOR INPUT AS FILE 7%
10034   DIM #7%,T0%(500%,2%)
10036   T0%=1%
10040   MAT C1%=CHI%:
        MAT C2%=CMI%:
        MAT C3%=CLO%:
        CLOSE 1%
10042   MAT N0%=ZER
10045!
        ! SET UP CELL-NET FILE
        !
10050   OPEN NNME$+".CLN" FOR OUTPUT AS FILE 3%
10060   DIM #3%,CLNTS(5000%)
10065!
        ! ADD ALL ENTRIES FOR ORIGINAL (NOT PARTIT-FORMED)
        !  CELLS
        !
10070   I%=1%:
```

CELNET

```
        J%=1%:
        K%=1%
10080   IF INP%(I%)=0% THEN J%=J%+1%:
                            GO TO 10120
10090   IF INP%(I%)=-1% THEN 10200
10100   INP=INP%(I%):
        J=J%:
        C(K%)=INP+J/1000
10110   K%=K%+1%
10120   I%=I%+1%
10130   GO TO 10080
10140!
        ! ADD ENTRIES FOR PARTIT-FORMED CELLS
        !
10200   I%=1%
10205   IF C1%(I%)=0% THEN 10300
10210   C1%=C2%(I%):
        C2%=C3%(I%)
10212   IF C1%(I%)=T0%(T0%,1%) THEN N0%(T0%(T0%,2%))=1%:
                                   T0%=T0%+1%:
                                   GO TO 10212
10220   FOR J%=1% TO K%-1%
10225           C0=C(J%)
10230           IF INT(C0)<>C1% THEN 10250
10232           IF N0%(INT((C0-C1%)*1000+.5)) THEN 10270
10234           C(K%)=C1%(I%)+C0-C1%:
                K%=K%+1%:
                GO TO 10270
10250           IF INT(C0)<>C2% THEN 10270
10252           IF N0%(INT((C0-C2%)*1000+.5)) THEN 10270
10254           C(K%)=C1%(I%)+C0-C2%:
                K%=K%+1%
10270   NEXT J%
10280   I%=I%+1%:
        GO TO 10205
10290!
        ! SORT CELL-NETS FOR EASIER ACCESS
        !
10300   C(K%)=0
10305   D%=K%-1%
10310   FLG%=0%:
        I%=1%
10315   IF C(I%+D%)=0 THEN 10330
10320   IF C(I%)>C(I%+D%) THEN T=C(I%):
                                   C(I%)=C(I%+D%):
                                   C(I%+D%)=T:
                                   FLG%=1%
10325   I%=I%+1%:
        GO TO 10315
10330   D%=D%/2%:
```

CELNET

```
        IF D% THEN 10310
10340   IF FLG% THEN D%=1%:
                    GO TO 10310
10400   I%=1%
10405!
        ! ELIMINATE DUPLICATES IN CELL-NET LIST
        !
10410   IF C(I%)=0 THEN 10500
10420   IF C(I%)==C(I%+1%) THEN 10422
                           ELSE I%=I%+1%: GO TO 10410
10422           C(J%)=C(J%+1%) FOR J%=I%+1% UNTIL C(J%)=0
10425   GO TO 10420
10500   K%=I%
10501   MAT PRINT C(K%)
10600           CLNTS(I%)=C(I%) FOR I%=1% TO K%
19999!
        ! ERROR HANDLER AND NORMAL PROGRAM EXIT
        !
20000   CLOSE 1,2,3
20010   END
```

```
LINEUP

1!
        ! LINEUP--- USES "FLIP-FLOP" ALGORITHM TO FORM
        !           LINEAR ORDERING OF CELLS FROM
        !           PARTITIONS
        !
20!
        ! VERSION 2
        !
30!
        ! MODIFICATION HISTORY
        !
40!
        ! 5-JAN-78              MODIFIED INITIALIZATION SEGMENT
        !                       TO ALLOW USE OF VERSION 2 OF CELNET
        !
999     EXTEND
1000    ON ERROR GO TO 20000
2000    DIM CLNTS(3000%),L%(500%),C1%(200%),
            C2%(200%),C3%(200%),UPNET%(200%),
            DWNNET%(200%),A%(200%),B%(200%)
9999!
        ! GET NETWORK NAME
        !
10000   INPUT "NET NAME";NNME$
10001!
        ! FETCH CELL-NETS FOR THIS NETWORK
        !
10005   OPEN NNME$+".CLN" FOR INPUT AS FILE 1%
10010   DIM #1%,C1(5000%)
10020   CLNTS(I%)=C1(I%) FOR I%=1% UNTIL C1(I%-1%)=0
10030   CLOSE 1
10050   LASTCELL%=1%:
        LASTCELL%=LASTCELL%+1% WHILE CLNTS(LASTCELL%)<>0
10055   LASTCELL%=LASTCELL%-1%
10059!
        ! FETCH PARTITIONS
        !
10060   OPEN NNME$+".PRT" FOR INPUT AS FILE 2%
10070   DIM #2%,CHI%(200%),CMI%(200%),CLO%(200%)
10080   MAT C1%=CHI%:
        MAT C2%=CMI%:
        MAT C3%=CLO%
10090   CLOSE 2
10092!
        ! FIND HIGHEST NUMBERED CELL
        !
10094   I%=1%:
        I%=I%+1% WHILE C1%(I%)<>0%
10096   C%=C1%(I%-1%)
10100!
```

LINEUP

```
        ! GET C1,C2 FOR HIGHEST #'D CELL
        !
10110   GO SUB 16000
10112   MAT L%=ZER
10115!
        !PUT C1,C2 IN L
        !
10120   L%(1%)=C1%:
        L%(2%)=C2%
10200!
        !********** MAIN LOOP **************
        !
        !
10210!
        ! FIND HIGHEST #'D CELL IN L
        !
10220   C%=0%:
        J%=0%
10230   FOR I%=1% UNTIL L%(I%)=0%
10240           IF L%(I%)>C% THEN J%=I%:
                        C%=L%(I%)
10250   NEXT I%
10252!
        ! IF ALL CELLS EXPANDED, QUIT
        !
10253   IF C%<500% THEN 11000
10254   REP%=J%
10255!
        ! FIND C1,C2 FOR THIS C
        !
10260   GO SUB 16000
10270!
        ! FORM UPNETS,DWNNETS
        !
10280   GO SUB 17000
10300!
        ! FORM LIST OF NETS THAT C1 IS IN (A ARRAY)
        ! AND LIST OF NETS THAT C2 IS IN (B ARRAY)
        !
10305   LA%=0%:
        LB%=0%
10310   I3%=1%:
        I3%=I3%+1% UNTIL INT(CLNTS(I3%))=C1%
10320   LA%=LA%+1%:
        A%(LA%)=INT((CLNTS(I3%)-INT(CLNTS(I3%)))*1000+.5)
10325   I3%=I3%+1%:
        IF INT(CLNTS(I3%))=C1% THEN 10320
10330   J3%=1%:
        J3%=J3%+1% UNTIL INT(CLNTS(J3%))=C2%
10340   LB%=LB%+1%:
```

```
LINEUP

        B%(LB%)=INT((CLNTS(J3%)-INT(CLNTS(J3%)))*1000+.5)
10350   J3%=J3%+1%:
        IF INT(CLNTS(J3%))=C2% THEN 10340
10400!
        ! COMPUTE J1,J2
        !
10410   J1%,J2%=0%
10420   FOR IA1%=1% TO LA%
10430           A%=A%(IA1%)
10435           FOR IA2%=1% UNTIL (UPNET%(IA2%)>A% OR UPNET%(IA2%)=0%)
10450                   IF UPNET%(IA2%)=A% THEN J1%=J1%+1%
10460           NEXT IA2%
10470           FOR IA3%=1% UNTIL (DWNNET%(IA3%)>A% OR DWNNET%(IA3%)=0%)
10490                   IF DWNNET%(IA3%)=A% THEN J1%=J1%-1%
10500           NEXT IA3%
10510   NEXT IA1%
10520   FOR JA1%=1% TO LB%
10530           B%=B%(JA1%)
10540           FOR JA2%=1% UNTIL (UPNET%(JA2%)>B% OR UPNET%(JA2%)=0%)
10560                   IF UPNET%(JA2%)=B% THEN J2%=J2%+1%
10570           NEXT JA2%
10580           FOR JA3%=1% UNTIL (DWNNET%(JA3%)>B% OR DWNNET%(JA3%)=0%)
10600                   IF DWNNET%(JA3%)=B% THEN J2%=J2%-1%
10610           NEXT JA3%
10620   NEXT JA1%
10700!
        ! USE J1,J2 TO GET C1,C2 ORDERING
        !
10710   IF J1%<J2% THEN T%=C2%:
                        C2%=C1%:
                        C1%=T%
10720!
        ! REPLACE EXPANDED CELL WITH C1,C2
        !
10730   L%(REP%)=C1%
10735   T1%=L%(REP%+1%)
10736   L%(REP%+1%)=C2%
10740   FOR IB1%=REP%+2% TO 200%
10750           IF L%(IB1%)=0% THEN L%(IB1%)=T1%:
                                    GO TO 10800
10760           T2%=L%(IB1%):
                L%(IB1%)=T1%:
                T1%=T2%
10770   NEXT IB1%
10800   GO TO 10200
10999!
        ! OUTPUT RESULTS AND SET UP X.LIN WITH LINEAR LIST
        !
11000   PRINT L%(I%) FOR I%=1% UNTIL L%(I%)=0%
11005   OPEN NNME$+".LIN" FOR OUTPUT AS FILE 3%
```

```
LINEUP

11006   DIM #3%,LIN%(200%)
11007   LIN%(I%)=L%(I%) FOR I%=1% TO 200%
11010   GO TO 20000
15999!
        ! FIND CELLS TO EXPAND C INTO
        !
16000!
        ! GET C1 & C2 FOR CELL #C
        !
16010   II%=1%:
        II%=II%+1% UNTIL C1%(II%)=C%:
        C1%=C2%(II%):
        C2%=C3%(II%)
16020   RETURN
17000!
        ! FORM UPNETS, DWNNETS FOR CELL AT POS. J
        !
17005   MAT UPNET%=ZER:
        MAT DWNNET%=ZER
17007   LUN%=0%:
        LDN%=0%
17008   IF J%=1% THEN 17100
17010   FOR I1%=1% TO J%-1%
17020           CELL%=L%(I1%)
17030           J1%=LASTCELL%
17040           J1%=J1%-1% UNTIL INT(CLNTS(J1%))=CELL%
17050           A7%=INT((CLNTS(J1%)-INT(CLNTS(J1%)))*1000+.5)
17055           UPNET%(A7%)=A7%
17060           J1%=J1%-1%:
                IF INT(CLNTS(J1%))=CELL% THEN 17050
17070   NEXT I1%
17100   IF L%(J%+1%)=0% THEN 17180
17110   FOR I2%=J%+1% UNTIL L%(I2%)=0%
17120           CELL%=L%(I2%)
17130           J2%=LASTCELL%
17140           J2%=J2%-1% UNTIL INT(CLNTS(J2 ))=CELL%
17150           B7%=INT((CLNTS(J2%)-INT(CLNTS(J2%)))*1000+.5)
17155           DWNNET%(B7%)=B7%
17160           J2%=J2%-1%:
                IF INT(CLNTS(J2%))=CELL% THEN 17150
17170   NEXT I2%
17180!
        ! ELIMINATE GAPS
        !
17190   FOR I4%=1% TO 200%
17200           IF UPNET%(I4%)<>0% THEN 17250
17210           FOR J4%=I4%+1% TO 200%
17220                   IF UPNET%(J4%)=0% THEN 17240
17230                   UPNET%(I4%)=UPNET%(J4%):
                        UPNET%(J4%)=0%:
```

LINEUP

```
                        GO TO 17250
17240           NEXT J4%
17245           GO TO 17255
17250   NEXT I4%
17252   PRINT "ERROR--NO SPACE IN UPNET":
        STOP
17255   LUN%=I4%
17300   FOR I6%=1% TO 200%
17310           IF DWNNET%(I6%)<>0% THEN 17360
17320           FOR J6%=I6%+1% TO 200%
17330                   IF DWNNET%(J6%)=0% THEN 17350
17340                   DWNNET%(I6%)=DWNNET%(J6%):
                        DWNNET%(J6%)=0%:
                        GO TO 17360
17350           NEXT J6%
17355           GO TO 17365
17360   NEXT I6%
17362   PRINT "ERROR--NO SPACE IN DWNNET":
        STOP
17365   LDN%=I6%
17450   RETURN
19999!
        ! ERROR HANDLER AND NORMAL PROGRAM EXIT
        !
20000   CLOSE 1,2,3,4
20010   PRINT TIME$(0)
20020   END
```

CROSS

```
1!
        ! CROSS--- FORMS X.CRS DATA BASE WHICH CONTAINS THE
        !          NUMBER OF NETS CROSSING EACH CELL BOUNDARY
        !          IN THE LINEAR CELL ORDERING. THIS DATA
        !          IS USED IN THE ROTATE PORTION OF FOLD
        !
1000    EXTEND
1010    ON ERROR GO TO 20000
10000   DIM L%(200%):
        DIM T%(50%):
        DIM CROSS(200%)
10005!
        ! GET NAME FOR THIS NETWORK
        !
10010   INPUT "NET NAME";NNME$
10015!
        ! FETCH NET LISTS
        !
10020   OPEN NNME$+".NCL" FOR INPUT AS FILE 1%
10030   DIM #1%,INP%(5000%)
10035!
        ! FETCH LINEAR ORDERING
        !
10040   OPEN NNME$+".LIN" FOR INPUT AS FILE 2%
10050   DIM #2%,L1%(200%)
10060   MAT L%=L1%:
        CLOSE 2
10065!
        ! FETCH PAD LIST
        !
10070   OPEN NNME$+".PAD" FOR INPUT AS FILE 3%
10080   DIM #3%,P1%(50%)
10090   MAT T%=P1%:
        CLOSE 3
10100!
        ! ELIMINATE PADS FROM LINEAR LIST
        !
10110   FOR I%=1% UNTIL T%(I%)=0%
10120           J%=1%:
                J%=J%+1% UNTIL L%(J%)=T%(I%)
10130           FOR K%=J% UNTIL L%(K%)=0%
10140                   L%(K%)=L%(K%+1%)
10150           NEXT K%
10160   NEXT I%
10170!
        ! CLEAR CROSSING ARRAY
        !
10180   MAT CROSS= ZER
10200!
        ! FIND # CROSSINGS AT EACH CELL BOUNDARY
```

CROSS

```
        !
10210   I%=1%
10220!
        ! CLEAR T ARRAY TO HOLD CURRENT NET
        !
10230   MAT T%=ZER
10232   K%=1%
10235!
        ! CHECK FOR END OF NETS
        !
10240   IF INP%(I%)=-1% THEN 10500
10250!
        ! CHECK FOR END OF THIS NET
        !
10260   IF INP%(I%)=0% THEN 10330
10270!
        ! ADD CELL TO T
        !
10271   T%=INP%(I%)
10272   J1%=1%:
        J1%=J1%+1% UNTIL (L%(J1%)=T% OR L%(J1%)=0%)
10274   IF L%(J1%)=0% THEN 10285
10280   T%(K%)=T%:
        K%=K%+1%
10285   I%=I%+1%
10287   GO TO 10240
10300!
        ! UPDATE CROSS ARRAY FOR THIS NET
        !
10305   IF K%<=2% THEN I%=I%+1%:
                       GO TO 10220
10310   J%=1%
10312!
        ! LOOK FOR CELL AT L%(J%) IN THIS NET
        !
10315   J1%=1%
10320   J1%=J1%+1% UNTIL (T%(J1%)=L%(J%) OR T%(J1%)=0%)
10330   IF T%(J1%)=0% THEN J%=J%+1%:
              GO TO 10315
10340!
        ! CELL AT L%(J%) IS FIRST OCCURRENCE OF CELLS
        !     IN THIS NET---ADD 1 TO EACH CROSSING UNTIL ALL
        !     CELLS IN THIS NET HAVE BEEN FOUND
        !
10350   K%=K%-1%
10360   K%=K%-1%:
        IF K%=0% THEN I%=I%+1%:
                        GO TO 10220
10370   CROSS(J%)=CROSS(J%)+1
10375   J%=J%+1%
```

CROSS

```
10380    J1%=1%:
         J1%=J1%+1% UNTIL (T%(J1%)=L%(J%) OR T%(J1%)=0%)
10390    IF T%(J1%)=0% THEN 10370
                        ELSE 10360
10500!
         ! SORT CROSSINGS IN DESCENDING ORDER
         !
10505    I%=1%
10510    I%=I%+1% UNTIL L%(I%)=0%:
         LAST%=I%-2%
10520    FOR I%=1% TO LAST%
10530             I=I%:
                  CROSS(I%)=CROSS(I%)+I/100
10540    NEXT I%
10550    FLAG%=0%
10560    FOR I%=1% TO LAST%-1%
10565             IF CROSS(I%)<CROSS(I%+1%) THEN 10580
10570             T=CROSS(I%):
                  CROSS(I%)=CROSS(I%+1%):
                  CROSS(I%+1%)=T:
                  FLAG%=1%
10580    NEXT I%
10600    IF FLAG% THEN 10550
10640!
         ! COPY CROSSINGS TO OUTPUT FILE
         !
10650    OPEN NNME$+".CRS" FOR OUTPUT AS FILE 4%
10655    DIM #4%,CROSS%(200%)
10660    FOR I%=1% TO LAST%
10670             C=CROSS(I%):
                  C=C-INT(C+.001):
                  CROSS%(I%)=INT(C*100+.5)
10680    NEXT I%
10690    CROSS%(LAST%+1%)=0%
10800    MAT PRINT CROSS%(LAST%)
19999!
         ! ERROR HANDLER AND NORMAL PROGRAM EXIT
         !
20000    CLOSE 1,2,3,4
20010    END
```

```
FOLD

1!
        ! FOLD-- LAYS LINEAR CELL ORDERING OUT ON CHIP
        !
20!
        ! VERSION 2 A
        !
30!
        ! MODIFICATION HISTORY
        !
40!
        ! 22-DEC-77       CHANGED ALG. FOR PLACEMENT OF CELL
        !                ON ROW I OR I+1 TO MAINTAIN C-SHAPES-- USE
        !                OF VARIABLE Z CHANGED TO LAST ROW
        !                (OF I,I+1) PLACED ON
        !
41!
        ! 30-DEC-77       MODIFIED OUTPUT FORMAT
        !
42!
        ! 11-JAN-78       AUTOMATED SELECTION AND MODIFICATION
        !                OF LOOKBACK DISTANCE-- ENABLED CONSTRUCTION
        !                OF XXX.FLD TO CONTAIN PLACEMENT
        !
1010    DIM C%(100%,100%)                !CHIP
1020    DIM L%(200%)                  !LINEAR PLACEMENT
1030    DIM W%(200%)                  !WIDTHS
1040    DIM P%(30%)                   !PADS
1050    DIM S%(100%)                  !SPACE LEFT ON ROWS
1060    DIM L0%(200%)
1070!
        ! GET CHIP DIMENSIONS (N=# CELL ROWS,
        !   M=# TRANSISTORS IN EACH ROW)
        !
1080    INPUT "CHIP DIMS(R,C)";N%,M%
1090!
        ! SET UP CHIP (C)
        !
1100    MAT C%=ZER(N%,M%)
1110!
        ! SET UP ARRAY S TO INDICATE SPACE REMAINING ON EACH ROW
        !
1120    MAT S%=ZER(N%)
1130!
        ! GET NAME FOR THIS NETWORK
        !
1140    INPUT "NET NAME";E$
1145    INPUT "OUTPUT TO";O$
1150!
        ! FETCH LINEAR ORDERING (IN ARRAY L%)
        !
```

```
FOLD

1160     OPEN E$+".LIN" FOR INPUT AS FILE 1%
1170     DIM #1%,L1%(200%)
1180     MAT L%=L1%:
         CLOSE 1
1190!
         ! FETCH CELL WIDTHS (IN ARRAY W%)
         !
1200     OPEN E$+".WID" FOR INPUT AS FILE 1%
1210     DIM #1%,W1%(200%)
1220     MAT W%=W1%:
         CLOSE 1
1230!
         ! FETCH PAD NUMBERS (IN ARRAY P%)
         !
1240     OPEN E$+".PAD" FOR INPUT AS FILE 1%
1250     DIM #1%,P1%(50%)
1260       P%(I%)=P1%(I%) FOR I%=1% TO 30%
1270     CLOSE 1
1280!
         ! FETCH CELL BOUNDARY CROSSING DATA (IN ARRAY C5%)
         !
1290     OPEN E$+".CRS" FOR INPUT AS FILE 2%
1300     DIM #2%, C5%(200%)
1310!
         ! C5% IS POINTER INTO ARRAY C5%
         !
1320     C5%=1%
1330     C%=1%:
         C%=C%+1% UNTIL L%(C%)=0%
1340!
         ! C1% IS NUMBER OF CELLS AND PADS
         ! C% IS NUMBER OF CELLS NOT PADS
         ! D% IS NUMBER OF PADS
         !
1350     C%=C%-1%
1360     C1%=C%
1370     IF P%(1%)=0% THEN 1550
1380     D%=1%:
         D%=D%+1% UNTIL P%(D%)=0%
1390     D%=D%-1%
1400     C%=C%-D%
1410!
         ! K6% IS NUMBER OF ROWS PRECEDING CURRENT ROW
         !  THAT WILL BE SCANNED FOR PLACING THE
         !  NEXT CELL
         !
1420     K6%=0%
1422     PRINT "INITIAL LOOKBACK DISTANCE IS";K6%
1430!
         ! ELIMINATE PADS FROM L% ARRAY
```

```
FOLD

        !
1440    FOR I%=1% TO D%
1450      FOR J%=1% TO 200%
1460            IF L%(J%)<>P%(I%) THEN 1510
1470            FOR K%=J% TO C1%+1%
1480              L%(K%)=L%(K%+1%)
1490            NEXT K%
1500            GO TO 1520
1510      NEXT J%
1520    NEXT I%
1530!
        ! ARRAY L0% IS ORIGINAL (UNROTATED) L% ARRAY
        !
1540    MAT L0%=L%
1545!
        ! LOAD S ARRAY--- S(I%) POINTS
        !   TO NEXT AVAILABLE LOCATION ON ROW I%-- IF
        ! S(I%)<0 THEN ROW IS RIGHT-TO-LEFT.
        ! OTHERWISE, ROW IS LEFT-TO-RIGHT
        !
1550    Z%=1%
1560    FOR I%=1% TO N%
1570      IF Z%=1% OR Z%=2% THEN S%(I%)=1%:GO TO 1590
1580      S%(I%)=-M%
1590      Z%=Z%+1%
1600      IF Z%<5% THEN 1620
1610      Z%=1%
1620    NEXT I%
1630!
        ! BEGIN PLACING CELLS
        !
1635!
        ! I% IS NUMBER OF ROW CURRENTLY USED AS BASE ROW
        !
1640    I%=1%                       !BASE ROW #
1645!
        ! J% IS POINTER INTO LINEAR ORDERING
        !
1650    J%=1%                       !POS IN L%
1655!
        ! Z% IS USED TO KEEP RECORD OF LAST ROW THAT A CELL WAS PLACED
        !   ON ( I OR I+1)
        !
1660    Z%=I%
1670    F1%=0%
1675!
        ! SCAN ROWS I%-K6% TO I%-1 TO SEE IF CURRENT CELL (L%(J%))
        !  CAN BE PLACED ON ONE OF THEM
        !
1680    IF I%-K6%<1% THEN E1%=1%
```

```
FOLD

              ELSE E1%=I%-K6%
1690      IF I%-1%<1% THEN 1750
            ELSE E2%=I%-1%
1700      FOR M5%=E1% TO E2%
1710        X%=M5%
1715!
            ! TRY TO PLACE CELL L%(J%) ON ROW X%
            !
1720        GO SUB 1890
1725!
            ! IF F%=1% THE CELL WAS PLACED ON ROW X%
            !
1730        IF F% THEN 1820
1740      NEXT M5%
1745!
          ! CELL WASNT PLACED BY LOOKBACK ROUTINE-- TRY TO PLACE
          !  IT ON ROW I OR ROW I+1
          !
1746!
          ! EQUAL SPACE ON ROWS I AND I+1 ?
          !
1750      IF S%(I%)<>S%(I%+1%) THEN 1775
1752!
          ! EQUAL SPACE REMAINS ON BOTH ROWS-- TRY TO PLACE CELL
          !   ON SAME ROW OF (I,I+1) AS THE LAST CELL (ROW Z)
          !
1755      X%=Z%:
          GO SUB 1890
1756!
          ! IF CELL WAS PLACED, GET NEXT CELL--
          !   OTHERWISE, TRY NEXT ROW
          !
1757      IF F% THEN 1830
              ELSE 1790
1770!
          ! SPACE REMAINING ON ROWS I AND I+1 IS NOT EQUAL--
          !  TRY TO PLACE CELL ON THE ONE OF THESE ROWS THAT HAS THE
          !  MOST SPACE REMAINING
          !
1775      IF S%(I%)>S%(I%+1%) THEN X%=I%+1%
                      ELSE X%=I%
1777      GO SUB 1890
1778!
          ! IF CELL WAS PLACED, GET NEXT CELL
          !
1780      IF F% THEN Z%=X%:
              GO TO 1830
1785!
          ! COULDNT PLACE CELL WITH THIS I%
          !
```

```
FOLD

1789!
      ! IF AT BOTTOM OF CHIP, CAN'T PLACE CELL AT ALL, GO TO ROTATE
      !   ROUTINE
      !
1790  IF I%=N% THEN 1860
1795!
      ! BUMP I% TO POINT TO NEXT ROW TO TRY
      !
1800  IF I%=N%-1% THEN I%=I%+1%
            ELSE I%=I%+2%
1810  GO TO 1660
1820!
1825!
      ! POINT J% TO NEXT CELL IN L%
      !
1830  J%=J%+1%
1835!
      ! IF ALL CELLS PLACED, QUIT
      !
1840  IF J%=C%+1% THEN PRINT "ALL PLACED --- ROTATION ";C5%-1%:
            GOTO 2220
1850  GO TO 1670                  !PLACE NEXT CELL
1860  !
1870  GO TO 2070
1890!
      ! THIS SUBROUTINE ATTEMPTS TO PLACE CELL L%(J%) ON
      !   ROW X%-- RETURNS F%=1% IF PLACED, F%=0% OTHERWISE
      !
1895!
      ! IF NO SPACE ON THIS ROW, NO USE IN TRYING
      !
1900  IF S%(X%)=0% THEN F%=0%:GO TO 2060
1910  Q%=L%(J%)                   !CELL #
1915!
      ! CHECK FOR L-TO-R OR R-TO-L ROW
      !
1920  IF S%(X%)<0% THEN 1990
1930  !L TO R ROW
1935!
      ! ROW X% IS L-TO-R, CHECK REMAINING SPACE
      !  TO SEE IF IT IS >= CELL WIDTH
      !  IF NOT, RESET F% AND RETURN
      !
1940  IF S%(X%)+W%(Q%)-1%>M% THEN F%=0%: GO TO 2060
1945!
      ! ROOM ENOUGH FOR CELL--- PLACE IT IN FIRST AVAILABLE SPACE
      !
1950  FOR K%=S%(X%) TO S%(X%)+W%(Q%)-1%          !PLACE CELL
1960    C%(X%,K%)=Q%
1970  NEXT K%
```

```
FOLD

1980    GO TO 2040
1990!
        !ROW X% IS R-TO-L, CHECK REMAINING SPACE TO
        !  SEE IF IT IS >= CELL WIDTH.  IF NOT,
        !  RESET F% AND RETURN
        !
2000    IF -S%(X%)-W%(Q%)+1%<1% THEN F%=0%:
              GO TO 2060
2005!
        ! ROOM ENOUGH FOR CELL--- PLACE IT IN FIRST AVAILABLE SPACE
        !
2010    FOR K%=-S%(X%) TO -S%(X%)-W%(Q%)+1% STEP -1%
2020      C%(X%,K%)=Q%
2030    NEXT K%
2035!
        ! UPDATE S ARRAY TO REFLECT NEW PLACED CELL
        !  SET F% FLAG
        !
2040    S%(X%)=S%(X%)+W%(Q%)
2050    F%=1%
2060    RETURN
2070!
        ! ROTATE ROUTINE--- USED WHEN THE LINEAR ORDER
        !  COULDNT BE PLACED ON THE CHIP--- SPLITS ORDER AT PLACE
        !  WITH FEWEST # BOUNDARY CROSSINGS--  EACH SUCEEDING SPLIT
        !  AT PLACES WITH MORE AND MORE CROSSINGS
        !
2080    IF C5%(C5%)<>0% THEN 2090
2081    K6%=K6%+1%:
        C5%=1%:
        IF K6%>N% THEN PRINT
          "CANT PLACE -- PARTIAL PLACEMENT FOLLOWS":GO TO 2230
2082    PRINT "LOOKBACK DISTANCE INCREASED TO";K6%
2083    GO TO 2200
2085!
        ! GET NEXT SPLIT BOUNDARY OUT OF CROSSING LIST,
        ! CLEAR L% AND REBUILD L% FROM L0% SPLIT AT C6%
        !
2090    C6%=C5%(C5%):
        K%=1%
2100    MAT L%=ZER
2110    FOR I%=C6%+1% UNTIL L0%(I%)=0%
2120      L%(K%)=L0%(I%):
          K%=K%+1%
2130    NEXT I%
2140    FOR I%=1% TO C6%
2150      L%(K%)=L0%(I%):
          K%=K%+1%
2160    NEXT I%
2170    C5%=C5%+1%
```

```
FOLD

2175!
        ! DUMP ROTATED ORDER
        !
2180!
2195!
        ! CLEAR CHIP & SPACE ARRAY--- START OVER
        !
2200    MAT C%=ZER:
        MAT S%=ZER
2210    GO TO 1550
2220!
        ! OUTPUT ROUTINE
        !
2230!
2240    OPEN O$ FOR OUTPUT AS FILE 3%
2241!
        ! SAVE CHIP IMAGE
        !
2242    OPEN E$+".FLD" FOR OUTPUT AS FILE 7%
2244    DIM #7%,C8%(2%),C7%(100%,100%)
2245    C8%(1%)=N%:
        C8%(2%)=M%
2248    C7%(I%,J%)=C%(I%,J%) FOR I%=1% TO N% FOR J%=1% TO M%
2249    CLOSE 7
2250    PRINT
2255    PRINT #3%,"LINEAR ORDER USED"
2257    PRINT #3%, L%(K%) FOR K%=1% UNTIL L%(K%)=0%
2259    PRINT #3%," " FOR K%=1% TO 5%
2260    IF M%<10% THEN K%=1%:
                GO TO 2360
2270    K%=1%
2280    FOR I%=1% TO N%
2290      PRINT #3%, TAB((J%-K%)*5%);C%(I%,J%); FOR J%=K% TO K%+9%
2300      PRINT #3%," "
2310      PRINT #3%," "
2320    NEXT I%
2330      PRINT #3%," " FOR J%=1% TO 10%
2340    K%=K%+10%
2350    IF K%+9%<M% THEN 2280
2360    FOR I%=1% TO N%
2370      PRINT #3%, TAB((J%-K%)*5%);C%(I%,J%); FOR J%=K% TO M%
2380      PRINT #3%," "
2390      PRINT #3%," "
2400    NEXT I%
2405!
        ! ERROR HANDLER AND NORMAL PROGRAM EXIT
        !
2410    CLOSE 3%
2420    END
        ! END OF PROGRAM
```

RATER

```
1!
        ! RATER-- RATES PLACEMENTS
        !
1000    DIM M%(200%,3%),T%(50%),R%(50%,2%)
10000   INPUT "NET NAME";N$
10001!
        ! FETCH CKT NETS
        !
10010   OPEN N$+".NCL" FOR INPUT AS FILE 1%
10020   DIM #1%,N%(5000%)
10021!
        ! FETCH PLACEMENT
        !
10030   OPEN N$+".FLD" FOR INPUT AS FILE 2%
10040   DIM #2%,C8%(2%),C7%(100%,100%)
10041!
        ! FROM PLACEMENT, SET UP M ARRAY SO THAT
        !       M(I,1)=ROW#, M(I,2)=LEFT-MOST COLUMN COVERED
        !       AND M(I,3)=RIGHT-MOST COLUMN COVERED FOR
        !       CELL NUMBER I
        !
10050   MAT M%=ZER:
        MAT R%=ZER
10060   FOR I%=1% TO C8%(1%)
10070           FOR J%=1% TO C8%(2%)
10080                   C%=C7%(I%,J%)
10090                   IF C%=0% THEN 10120
10100                   IF M%(C%,1%)=0% THEN M%(C%,1%)=I%:
                                             M%(C%,2%)=J%:
                                             M%(C%,3%)=J%-1%
10110                   M%(C%,3%)=M%(C%,3%)+1%
10120           NEXT J%
10130   NEXT I%
10131!
        ! FOR NET IN NET LIST, ENTER ALL CELLS
        !       (WITHOUT PADS) IN T ARRAY
        !
10140   U1%=1%
10145   T1%=0%
10150   IF N%(U1%)=0% THEN 10200
10160   IF N%(U1%)=-1% THEN 10400
10170   IF M%(N%(U1%),1%)=0% THEN 10190
10180   T1%=T1%+1%:
        T%(T1%)=N%(U1%)
10190   U1%=U1%+1%:
        GO TO 10150
10191!
        ! FIND IF THIS NET IS STRAIGHT BY:
        !       1. STRAIGHT IF ALL CELLS ON SAME CELL ROW
        !       2. STRAIGHT IF LEFT-MOST RIGHT-MOST COL
```

RATER

```
        !           LIES RIGHT OF RIGHT-MOST LEFT-MOST
        !           COL
        !
10200   IF T1%<2% THEN 10190
10205   M0%=0%:
        M1%=999%
10206   R%(T1%,1%)=R%(T1%,1%)+1%
10210   FOR I%=1% TO T1%
10230           N%=T%(I%)
10240           IF I%=1% THEN M2%=M%(N%,1%):
                                GO TO 10260
10250           IF M%(N%,1%)<>M2% THEN M2%=0%
10260           IF M%(N%,2%)>M0% THEN M0%=M%(N%,2%)
10270           IF M%(N%,3%)<M1% THEN M1%=M%(N%,3%)
10280           IF M0%>M1% THEN
                                IF M2%=0% THEN 10145
10290   NEXT I%
10300   R%(T1%,2%)=R%(T1%,2%)+1%
10310   GO TO 10145
10311!
        ! DUMP RESULTS
        !
10400   PRINT "NET SIZE";TAB(20);"NUMBER NETS";TAB(45);"% STRAIGHT"
10405   R3=0:
        R4=0
10410   FOR I%=1% TO 50%
10420           IF R%(I%,1%)=0% THEN 10450
10430           R1=R%(I%,1%):
                R2=R%(I%,2%):
                P=R2/R1*100:
                R3=R3+R1:
                R4=R4+R2
10440           PRINT I%;TAB(20);R1;TAB(45);P:
                PRINT
10450   NEXT I%
10460   PRINT:
        PRINT "TOTAL # NETS ANALYSED";R3:
        PRINT:
        PRINT " # STRAIGHT";R4:
        PRINT:
        PRINT " % STRAIGHT";R4/R3*100
10470   END
```

```
LITLAY

1!
        ! LITLAY -- FORMS REDUCED CHIP IMAGE FOR HAND ROUTING
        !
        !           24 FEB 78
        !
10!
        ! MODIFICATION HISTORY
        !
10000   ON ERROR GOTO 11880
10020   EXTEND
10040   DIM CHIP%(100%,100%),NUMNETS%(200%),START%(200%)
10050!
        ! BUILD SCRATCH FILE
        !
10060   OPEN "HELP.TMP" FOR OUTPUT AS FILE 5
10080   DIM #5,T%(50%,500%)
10100   T%(50%,500%)=0
10110!
        ! GET INSTRUCTIONS
        !
10120   INPUT "NET NAME",NAME.$
10140   INPUT "PAPER WIDTH",LINE.WID%
10160   INPUT "OUTPUT TO <KB:>",OUTPUT$\
        IF OUTPUT$="" THEN OUTPUT$="KB:"
10180   INPUT "HEADER <NET NAME>",HEADER$
10200   IF HEADER$="" THEN
                HEADER$=NAME.$
10210!
        ! FETCH CHIP
        !
10220   OPEN NAME.$+".FLD" FOR INPUT AS FILE 1
10240   DIM #1%,C8%(2%),C7%(100%,100%)
10260   NUMROWS%=C8%(1%)\
        NUMCOLS%=C8%(2%)\
        MAT CHIP%=ZER(NUMROWS%,NUMCOLS%)
10280   CHIP%(I%,J%)=C7%(I%,J%)
                FOR I%=1% TO NUMROWS% FOR J%=1% TO NUMCOLS%
10300   CLOSE 1
10310!
        ! FETCH CELL-NETS
        !
10320   OPEN NAME.$+".CLN" FOR INPUT AS FILE 2
10340   DIM #2, CELNET(5000%)
10350!
        ! FETCH CELL WIDTHS
        !
10360   OPEN NAME.$+".WID" FOR INPUT AS FILE 1
10380   DIM #1,WIDTH%(200%)
10400   MAT NUMNETS%=ZER\
        MAT START%=ZER
```

```
LITLAY

10420!
        ! FIND NUMNETS, START FOR EACH CELL
        !
10440   FOR I%=1% WHILE INT(CELNET(I%))<500%
10460           NUMNETS%(INT(CELNET(I%)))=
                        NUMNETS%(INT(CELNET(I%)))+1%\
                IF START%(INT(CELNET(I%)))=0% THEN
                        START%(INT(CELNET(I%)))=I%
10480   NEXT I%
10500!
        ! SET TRANSISTOR WIDTH
        !
10520   TRANS.WID%=0%
10540   FOR I%=1% WHILE INT(CELNET(I%))<500%
10560           IF WIDTH%(INT(CELNET(I%)))=0% THEN 10600
10580           T%=INT(CELNET(I%))\
                TRY%=INT((2*(NUMNETS%(T%)+1%))/WIDTH%(T%)+.99)\
                IF TRY%>TRANS.WID% THEN TRANS.WID%=TRY%
10600   NEXT I%
10620   IF TRANS.WID%<2% THEN TRANS.WID%=2%
10640!
        ! SET # TRANS / LINE
        !
10660   TRANS.PER.LINE%=1%
10680   IF TRANS.PER.LINE%*((TRANS.WID%+1%))+1%<LINE.WID% THEN
                        TRANS.PER.LINE%=TRANS.PER.LINE%+1%\
                        GOTO 10680
10700!
        ! ADJUST LINE WIDTH
        !
10720   TRANS.PER.LINE%=TRANS.PER.LINE%-1%
10740   LINE.WID%=TRANS.PER.LINE%*(TRANS.WID%+1%)+1%
10750!
        ! OPEN OUTPUT DEVICE
        !
10760   OPEN OUTPUT$ AS FILE 7
10770!
        ! CLEAR NET POINTER
        !
10780   NET.INDEX%=0%
10790!
        ! CLEAR OUTPUT TEMPLATE
        !
10800   MAT T%=ZER(NUMROWS%,((TRANS.WID%+1%)*NUMCOLS%)+10%)
10810!
        ! FORM L-H EDGE OF CHIP
        !
10820   T%(I%,1%)=1% FOR I%=1% TO NUMROWS%
10830!
        ! FORM TRANSISTOR BOUNDARIES
```

```
LITLAY

        !
10840   T%=1%\
        C%=0%
10860   T%=T%+TRANS.WID%+1%
10880   I%=T%
10900   T%(J%,I%)=3% FOR J%=1% TO NUMROWS%
10920   C%=C%+1%
10940   IF C%<NUMCOLS% THEN 10860
10950!
        ! INITIALIZE CHIP POINTER TO (1,1)
        !
10960   ROW%=1%
10980   COL%=1%\
        I%=2%
10990!
        ! LOAD CELL NUMBER INTO TEMPLATE
        !
11000   CELL.NUM%=CHIP%(ROW%,COL%)\
        T%(ROW%,I%)=CELL.NUM%+1000%\
        T%(ROW%,I%+1%)=2%
11010!
        ! CLEAR NET POINTER AND BUMP TEMPLATE POINTER
        !
11020   NET.INDEX%=0%
11040   I%=I%+1%
11050!
        ! IF LOOKING AT TRANS BOUNDARY, GO TO NEXT TRANS ROUTINE
        !
11060   IF T%(ROW%,I%)=3% THEN 11180
11070!
        ! IF NO SPACE BETWEEN HERE AND LAST NUMBER, SKIP TO NEXT LOC
        !
11080   IF T%(ROW%,I%-1%)>=1000% THEN 11040
11085   IF CELL.NUM%=0% THEN 11040
11090!
        ! GET NEXT CELL-NET
        !
11100   NET=CELNET(START%(CELL.NUM%)+NET.INDEX%)
11110!
        ! IF NET NOT FOR THIS CELL, SKIP
        !
11120   IF INT(NET)<>CELL.NUM% THEN 11040
11130!
        ! LOAD NET # INTO TEMPLATE
        !       AND GO TO NEXT COL
        !
11140   T%(ROW%,I%)=INT((NET-CELL.NUM%)*1000+.5)+1000%
11160   NET.INDEX%=NET.INDEX%+1%\
        GO TO 11040
11170!
```

```
LITLAY

        ! NEXT TRANSISTOR ROUTINE
        !
11175!
        ! BUMP CHIP COL PNTR, TEMPLATE PNTR
        !
11180   COL%=COL%+1%\
        I%=I%+1%
11190!
        ! IF ALL CHIP COLS TREATED FOR THIS ROW,
        !       LOAD R-H EDGE AND TRY NEXT CHIP ROW
        !
11200   IF COL%>NUMCOLS% THEN T%(ROW%,I%-1%)=1%\
                              GOTO 11260
11210!
        ! IF STILL IN SAME CELL, CONTINUE
        !
11215   IF CHIP%(ROW%,COL%)=0% THEN 11240
11220   IF CHIP%(ROW%,COL%)=CELL.NUM% THEN 11000
11230!
        ! NEW CELL--- DRAW CELL BOUNDARY AND TREAT NEW CELL
        !
11240   T%(ROW%,I%-1%)=1%\
        GOTO 11000
11250!
        ! BUMP CHIP ROW POINTER
        !       IF STILL ON CHIP, HANDLE NEW ROW
        !
11260   ROW%=ROW%+1%\
        IF ROW%<=NUMROWS% THEN 10980
11270!
        ! LAST ROW TREATED, SAVE POS OF END OF ROW IN END.%
        !
11280   END.%=I%-1%
11290!
        ! SET BASE, ROW FOR PRINTOUT
        !
11300   BASE%=1%
11320   ROW%=1%
11330!
        ! PRINT PAGE HEADER
        !
11340   GOSUB 11780
11350!
        ! IF THIS IS LAST PAGE, ADJUST LINE WIDTH  TO ACTUAL
        !       WIDTH REMAINING
        !
11360   IF END.%<BASE%+LINE.WID% THEN LINE.WID%=END.%-BASE%+2%
11370!
        ! PRINT TOP OF CELL ROW
        !
```

```
LITLAY

11380    PRINT #7,STRING$(LINE.WID%-1%,ASCII("#"))
11390!
         ! CLEAR CELL ROW STRINGS
         !
11400    A$=""\
         B$=""\
         C$=""
11420    FOR COL%=BASE% TO BASE%+LINE.WID%-2%
11440         T%=T%(ROW%,COL%)
11450!
              ! ADD APPROP. NON-NUMERIC CHARS TO STRINGS
              !
11460         IF T%>=1000% THEN 11600
11480         IF T%=0% THEN D$=" "
11500         IF T%=1% THEN D$="#"
11520         IF T%=2% THEN D$="-"
11540         IF T%=3% THEN D$=":"
11560         A$=A$+D$\
              B$=B$+D$\
              C$=C$+D$
11580         GOTO 11620
11590!
              ! NUMBER ENCOUNTERED IN TEMPLATE-- CONVERT TO
              !       CHARS AND ADD ONE CHAR TO EACH STRING
              !
11600         P$=NUM$(T%-1000%)\
              P$=CVT$$(P$,6%)\
              P$=P$+" " WHILE LEN(P$)<5%\
              A$=A$+MID(P$,1,1)\
              B$=B$+MID(P$,2,1)\
              C$=C$+MID(P$,3,1)
11620    NEXT COL%
11630!
         ! PRINT CELL ROW STRINGS
         !
11640    PRINT #7,A$\
         PRINT #7,B$\
         PRINT #7,C$
11650!
         ! BUMP TEMPLATE ROW PNTR
         !
11660    ROW%=ROW%+1%\
         IF ROW%<=NUMROWS% THEN 11360
11670!
         ! ALL ROWS HANDLED-- PRINT CHIP BOTTOM
         !
11680    PRINT #7,STRING$(LINE.WID%-1%,ASCII("#"))
11690!
         ! SET UP POINTERS FOR NEXT PAGE
         !
```

LITLAY

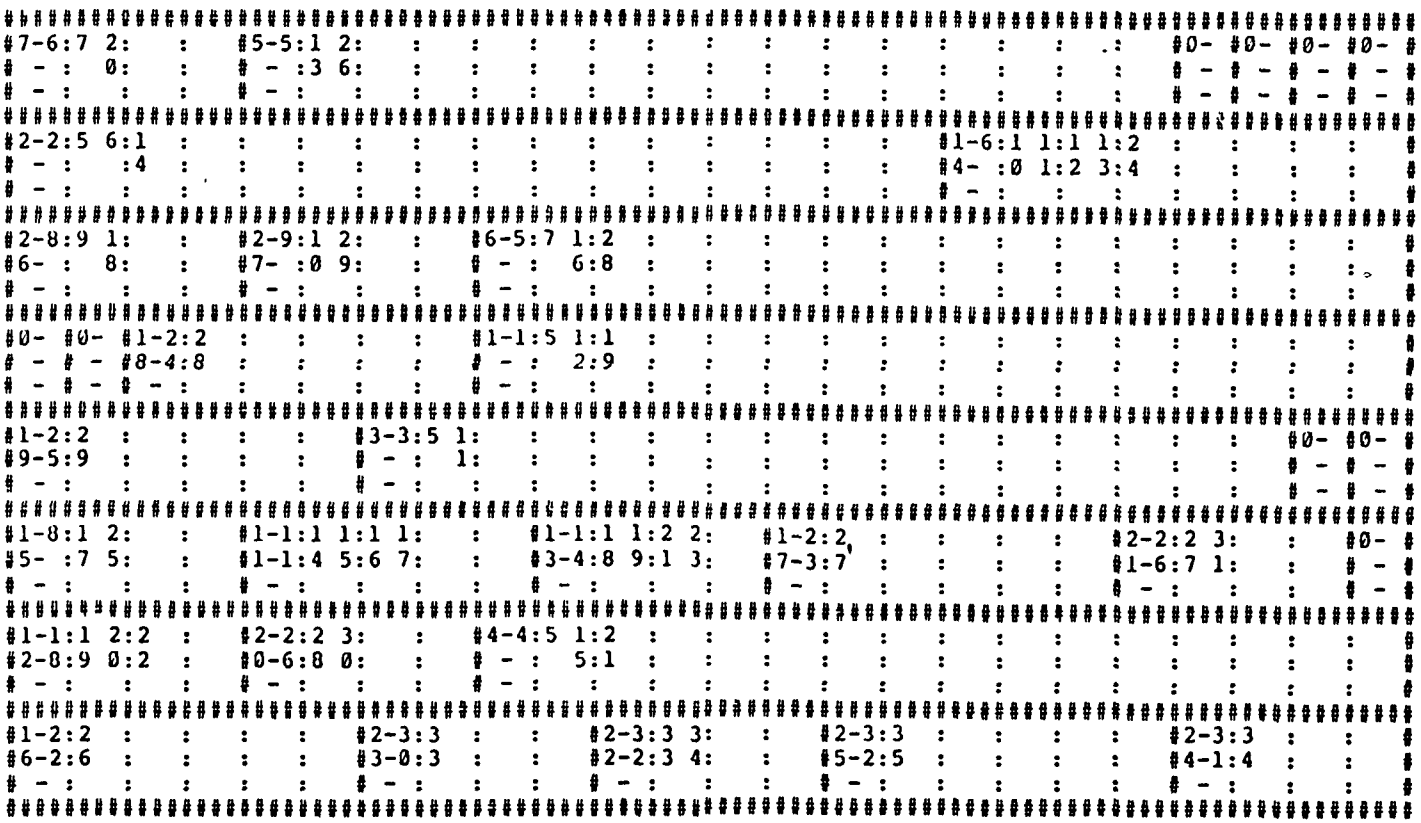
```
11700   BASE%=COL%+1%\
        IF BASE%>(TRANS.WID%*(NUMCOLS%+1%)+1%) THEN 11880
11720   GOTO 11320
11770!
        ! HEADER PRINTING
        !
11780   IF BASE%>1% THEN PRINT #7,CHR$(12%)
11800           PRINT #7," " FOR I=1 TO 3
11820   PRINT #7,HEADER$+" AT "+TIME$(0)+" ON "+DATE$(0)
11840           PRINT #7," " FOR I%=1% TO 3%
11860   RETURN
11870!
        ! ERROR PROCESSOR
        !
11880   CLOSE 1,2,7
11900   PRINT ERR,ERL
11920   END
```

APPENDIX B

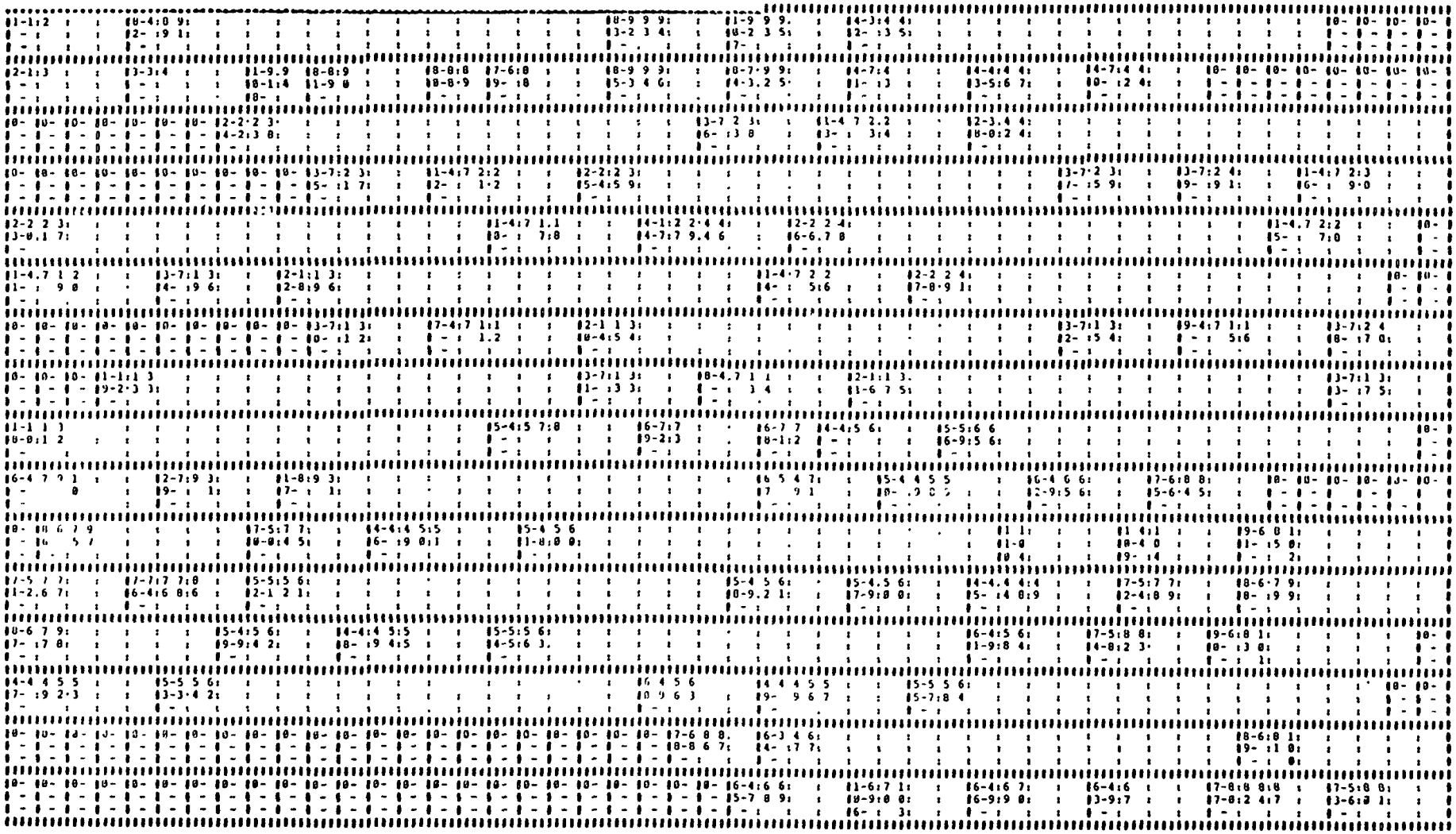
EXAMPLE PLACEMENTS

System-produced placements for the NAS101, NAS15, and SAUCKT networks are shown in this section.

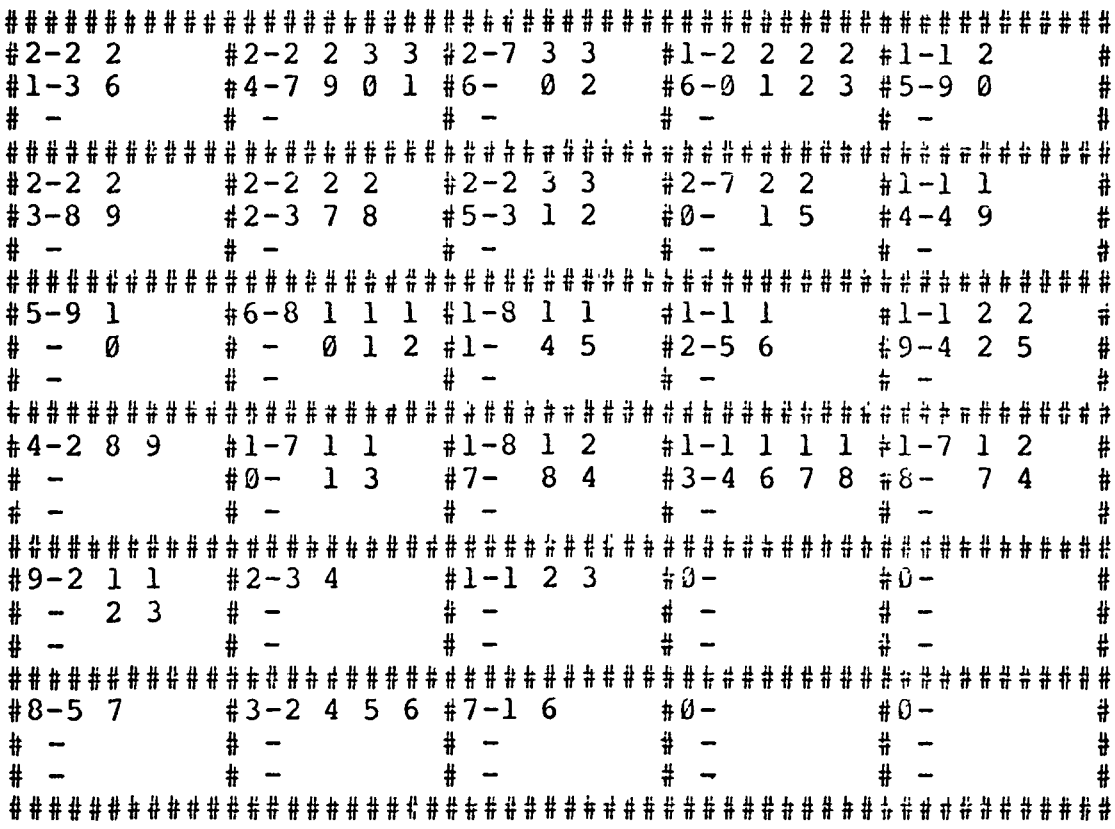
NAS101 NETWORK LAYOUT



NAS15 NETWORK LAYOUT



SAUCKT NETWORK LAYOUT



☆ U.S. GOVERNMENT PRINTING 1979—640-081/436 REGION NO. 4